FUJITSU SEMICONDUCTOR

CONTROLLER MANUAL

CM25-10146-1E

F^2MC-8L

8-BIT MICROCONTROLLER

MB89950/950A Series HARDWARE MANUAL

FUJITSU

F^2MC-8L

8-BIT MICROCONTROLLER

MB89950/950A Series

HARDWARE MANUAL

FUJITSU LIMITED

PREFACE

■ Objectives and Intended Reader

The MB89950/950A series has been developed as a general-purpose version of the F^2MC-8L family consisting of proprietary 8-bit, single-chip microcontrollers. The MB89950/950A series is applicable to a wide range of applications from consumer products to industrial equipment, including portable devices.

This manual describes the functions and operation of the MB89950/950A series and is aimed at engineers using the MB89950/950A series of microcontrollers to develop actual products. See the F^2MC-8L MB89600 Series Programming Manual for details on the MB89950/950A instruction set.

■ Trademarks

F^2MC is the abbreviation of FUJITSU Flexible Microcontroller.

Other system and product names in this manual are trademarks of respective companies or organizations.

The symbols TM and ® are sometimes omitted in this manual.

■ Structure of This Manual

This manual contains the following 12 chapters:

CHAPTER 1 "OVERVIEW"

This chapter describes the main features and basic specifications of the MB89950/950A series.

CHAPTER 2 "HANDLING DEVICE"

This chapter describes points to note when using the general-purpose single-chip microcontroller.

CHAPTER 3 "CPU"

This chapter describes the functions and operation of the CPU.

CHAPTER 4 "I/O PORTS"

This chapter describes the functions and operation of the I/O ports.

CHAPTER 5 "TIMEBASE TIMER"

This chapter describes the functions and operation of the timebase timer.

CHAPTER 6 "WATCHDOG TIMER"

This chapter describes the functions and operation of the watchdog timer.

CHAPTER 7 "8-BIT PWM TIMER"

This chapter describes the functions and operation of the 8-bit PWM timer.

CHAPTER 8 "PULSE WIDTH COUNT TIMER (PWC)"

This chapter describes the functions and operation of the pulse width count timer (PWC).

CHAPTER 9 "8-BIT SERIAL I/O"

This chapter describes the functions and operation of the 8-bit serial I/O.

CHAPTER 10 "UART"

This chapter describes the functions and operation of the UART.

CHAPTER 11 "EXTERNAL INTERRUPT CIRCUIT (EDGE)"

This chapter describes the functions and operation of the external interrupt circuit.

CHAPTER 12 "LCD CONTROLLER/DRIVER"

This chapter describes the functions and operation of the LCD controller/driver.

APPENDIX

This appendix includes I/O maps, instruction lists, and other information.

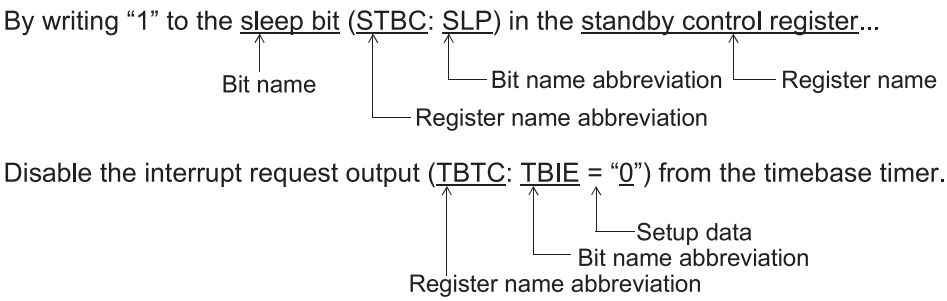
- The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.
- The information and circuit diagrams in this document are presented as examples of semiconductor device applications, and are not intended to be incorporated in devices for actual use. Also, FUJITSU is unable to assume responsibility for infringement of any patent rights or other rights of third parties arising from the use of this information or circuit diagrams.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
 Please note that Fujitsu will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.

©2002 FUJITSU LIMITED Printed in Japan

READING THIS MANUAL

■ Notations of the Register Name and Pin Name

● Example for description of register name and bit name



● Notations of a double-purpose pin

P22/SCK pin

Some pins can be used by switching their functions using, for example, settings by a program. Each double-purpose pin is represented by separating the name of each function using "/".

■ Documents and Development Tools Required for Development

Items necessary for the development of this product are as follows.

To obtain the necessary documents and development tools, contact a company sales representative.

● Manuals required for development

[Check field]

- ☐ F^2MC-8L MB89950/950A series data sheet (provides a table of electrical characteristics and various examples of this product)
- ☐ F^2MC-8L Programming Manual (manual including instructions for the F^2MC-8L family)
- * FR/F^2MC Family Softune C Compiler Manual (required only if C language is used for development) (manual describing how to develop and activate programs in the C language)
- * FR/F^2MC Family Softune Assembler Manual for V3 (manual describing program development using the assembler language)
- * FR/F^2MC Family Softune Linkage Kit Manual for V3 (manual describing functions and operations of the assembler, linker, and library manager

Manuals with the * mark are attached to each product.

Other manuals, such as those for development, are attached to respective products.

Software required for development

[Check field]

- ☐ Softune V3 Workbench
- ☐ Softune V3 for personal ICE (required only if the evaluation is performed for the personal-ICE)
- ☐ Softune V3 for compact ICE (required only if the evaluation is performed for the compact-ICE)

The type of software product is dependent on the OS to be used.

For details, see the F^2MC Development Tool Catalog or Product Guide.

❍ **What is needed for evaluation on the one-time PROM microcomputer (if the programming operation is performed at your side)**

[Check field]

- ☐ MB89P955
- ☐ EPROM programmer
 (Programmer available for the MBM27C1001)
- ☐ Package conversion adapter
 ROM-64QF2-28DP-8L3

● Development tools

[Check field]

- ☐ MB89PV950 (piggyback/evaluation device)
- ☐ Development tool

Main unit	Pod	Probe
MB2141A + MB2144-505		MB2144-203

To use a the other development environment, contact respective makers.

● References

- "F^2MC Development Tool Catalog"
- "Microcomputer Product Guide"

CONTENTS

CHAPTER 1 OVERVIEW 1
1.1 MB89950/950A Series Features 2
1.2 MB89950/950A Series Product Range 4
1.3 Differences among Products 6
1.4 Block Diagram of MB89950/950A Series 7
1.5 Pin Assignment 8
1.6 Package Dimensions 10
1.7 I/O Pins and Pin Functions 12

CHAPTER 2 HANDLING DEVICES 17
2.1 Notes on Handling Devices 18

CHAPTER 3 CPU 21
3.1 Memory Space 22
3.1.1 Special Areas 24
3.1.2 Storing 16-bit Data in Memory 26
3.2 Dedicated Registers 27
3.2.1 Condition Code Register (CCR) 29
3.2.2 Register Bank Pointer (RP) 32
3.3 General-purpose Registers 33
3.4 Interrupts 35
3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3) 36
3.4.2 Interrupt Processing 37
3.4.3 Multiple Interrupts 39
3.4.4 Interrupt Processing Time 40
3.4.5 Stack Operation during Interrupt Processing 41
3.4.6 Stack Area for Interrupt Processing 42
3.5 Resets 43
3.5.1 External Reset Pin 45
3.5.2 Reset Operation 46
3.5.3 Pin States during Reset 48
3.6 Clocks 49
3.6.1 Clock Generator 51
3.6.2 Clock Controller 53
3.6.3 Oscillation Stabilization Delay Time 55
3.7 Standby Mode (Low-power Consumption) 57
3.7.1 Operating States in Standby Mode 58
3.7.2 Sleep Mode 59
3.7.3 Stop Mode 60
3.7.4 Standby Control Register (STBC) 61
3.7.5 State Transition Diagram 63
3.7.6 Notes on Using Standby Mode 65
3.8 Memory Access Mode 67

CHAPTER 4 I/O PORTS 69
4.1 Overview of I/O Ports 70
4.2 Port 0 72
4.2.1 Port 0 Data Register (PDR0) 74
4.2.2 Operation of Port 0 75
4.3 Port 1 77
4.3.1 Port 1 Data Register (PDR1) 79
4.3.2 Operation of Port 1 80
4.4 Port 2 82
4.4.1 Port 2 Data Register (PDR2) 84
4.4.2 Operation of Port 2 85
4.5 Port 3 86
4.5.1 Port 3 Data Register (PDR3) 89
4.5.2 Operation of Port 3 90
4.6 Port 4 92
4.6.1 Port 4 Registers (PDR4, DDR4) 94
4.6.2 Operation of Port 4 96
4.7 Program Example for I/O Ports 98

CHAPTER 5 TIMEBASE TIMER 99
5.1 Overview of Timebase Timer 100
5.2 Block Diagram of Timebase Timer 102
5.3 Timebase Timer Control Register (TBTC) 104
5.4 Timebase Timer Interrupt 106
5.5 Operation of Timebase Timer 107
5.6 Notes on Using Timebase Timer 109
5.7 Program Example for Timebase Timer 110

CHAPTER 6 WATCHDOG TIMER 111
6.1 Overview of Watchdog Timer 112
6.2 Block Diagram of Watchdog Timer 113
6.3 Watchdog Timer Control Register (WDTC) 115
6.4 Operation of Watchdog Timer 116
6.5 Notes on Using Watchdog Timer 118
6.6 Program Example for Watchdog Timer 119

CHAPTER 7 8-BIT PWM TIMER 121
7.1 Overview of 8-bit PWM Timer 122
7.2 Block Diagram of 8-bit PWM Timer 124
7.3 Structure of 8-bit PWM Timer 126
7.3.1 PWM Control Register (CNTR) 128
7.3.2 PWM Compare Register (COMR) 130
7.4 8-bit PWM Timer Interrupts 131
7.5 Operation of Interval Timer Function 132
7.6 Operation of PWM Timer Function 134
7.7 States in Each Mode during 8-bit PWM Timer Operation 135
7.8 Notes on Using 8-bit PWM Timer 137

7.9 Program Example for 8-bit PWM Timer 138

CHAPTER 8 PULSE WIDTH COUNT TIMER (PWC) 141

8.1 Overview of Pulse Width Count Timer 142
8.2 Block Diagram of Pulse Width Count Timer 144
8.3 Structure of Pulse Width Count Timer 146
8.3.1 PWC Pulse Width Control Register 1 (PCR1) 148
8.3.2 PWC Pulse Width Control Register 2 (PCR2) 150
8.3.3 PWC Reload Buffer Register (RLBR) 152
8.3.4 PWC Noise Filter Control Register (NCCR) 154
8.4 Pulse Width Count Timer Interrupts 155
8.5 Operation of Interval Timer Function 156
8.6 Operation of Pulse Width Measurement Function 159
8.7 Operation of Noise Filter Circuit 162
8.8 States in Each Mode during Pulse Width Count Timer Operation 163
8.9 Notes on Using Pulse Width Count Timer 164
8.10 Program Example for Timer Function of Pulse Width Count Timer 165

CHAPTER 9 8-BIT SERIAL I/O 169

9.1 Overview of 8-bit Serial I/O 170
9.2 Block Diagram of 8-bit Serial I/O 171
9.3 Structure of 8-bit Serial I/O 173
9.3.1 Serial Mode Register (SMR) 176
9.3.2 Serial Data Register (SDR) 179
9.4 8-bit Serial I/O Interrupts 180
9.5 Operation of Serial Output 181
9.6 Operation of Serial Input 183
9.7 States in Each Mode during 8-bit Serial I/O Operation 185
9.8 Notes on Using 8-bit Serial I/O 188
9.9 Connection Example for 8-bit Serial I/O 189
9.10 Program Example for 8-bit Serial I/O 190

CHAPTER 10 UART 193

10.1 Overview of UART 194
10.2 Structure of UART 199
10.3 UART Pins 202
10.4 UART Registers 204
10.4.1 Serial Mode Control Register 1 (SMC1) 205
10.4.2 Serial Rate Control Register (SRC) 207
10.4.3 Serial Status and Data Register (SSD) 209
10.4.4 Serial Input Data Register (SIDR) 211
10.4.5 Serial Output Data Register (SODR) 212
10.4.6 Serial Mode Control Register 2 (SMC2) 213
10.5 UART Interrupts 215
10.6 Operation of UART 216
10.7 Operation of Mode 0, 1, 3 217
10.8 Program Example for UART 220

CHAPTER 11 EXTERNAL INTERRUPT CIRCUIT (EDGE) 223
11.1 Overview of the External Interrupt Circuit 224
11.2 Block Diagram of the External Interrupt Circuit 225
11.3 Structure of the External Interrupt Circuit 226
11.3.1 External Interrupt Control Register (EIC) 228
11.4 External Interrupt Circuit Interrupts 230
11.5 Operation of the External Interrupt Circuit 231
11.6 Program Example for the External Interrupt Circuit 232

CHAPTER 12 LCD CONTROLLER/DRIVER 233
12.1 Overview of LCD Controller/Driver 234
12.2 Block Diagram of LCD Controller/Driver 235
12.2.1 LCD Controller/Driver Internal Voltage Divider 237
12.2.2 LCD Controller/Driver External Voltage Divider 239
12.3 Structure of LCD Controller/Driver 241
12.3.1 LCD Control Register (LCDR) 244
12.3.2 Segment Output Select Register (SEGR) 246
12.3.3 Display RAM 248
12.4 Operation of LCD Controller/Driver 250
12.4.1 Output Waveforms during LCD Controller/Driver Operation (1/2 Duty Ratio) 251
12.4.2 Output Waveforms during LCD Controller/Driver Operation (1/3 Duty Ratio) 254
12.4.3 Output Waveforms during LCD Controller/Driver Operation (1/4 Duty Ratio) 257
12.5 Program Example for LCD Controller/Driver 260

APPENDIX 263
APPENDIX A I/O Map 264
APPENDIX B Overview of Instructions 266
B.1 Overview of F^2MC-8L Instructions 267
B.2 Addressing 269
B.3 Special Instructions 274
B.4 Bit Manipulation Instructions (SETB, CLRB) 278
B.5 F^2MC-8L Instructions 279
B.6 Instruction map 286
APPENDIX C Mask Options 287
APPENDIX D Programming Specifications for One-Time PROM And EPROM Microcontroller 289
D.1 Programming Specifications for One-time PROM and EPROM Microcontrollers 290
D.2 Programming Yield and Erasure 293
D.3 Programming to the EPROM with Piggyback/Evaluation Device 294
APPENDIX E MB89950/950A Series Pin States 295

INDEX 297

CHAPTER 1
OVERVIEW

This chapter describes the main features and basic specifications of the MB89950/950A series.

1.1 "MB89950/950A Series Features"

1.2 "MB89950/950A Series Product Range"

1.3 "Differences among Products"

1.4 "Block Diagram of MB89950/950A Series"

1.5 "Pin Assignment"

1.6 "Package Dimensions"

1.7 "I/O Pins and Pin Functions"

1.1 MB89950/950A Series Features

The MB89950/950A series is a line of the general-purpose, single-chip microcontrollers. In addition to a compact instruction set, the microcontrollers contain a variety of peripheral functions such as an LCD controller/driver, UART, a serial I/O, PWC timer, PWM timer and external interrupts.

■ MB89950/950A series features

- Various package options
 - QFP packages (0.65 mm lead pitch) for MB89951A/MB89953A/MB89P955 only
- High speed processing at low voltage

 Minimum execution time: 0.8 µs/5 MHz
- F^2MC-8L family CPU core

 Instruction set optimized for controllers
 - Multiplication and division instructions
 - 16-bit arithmetic operations
 - Test and branch instructions
 - Bit manipulation instructions, etc.
- Single-clock control system
 - Main clock: max. 5 MHz
- Four types of timer
 - 21-bit timebase timer
 - Watchdog timer
 - 8-bit PWM timer (also can be used as an interval timer)
 - 8-bit PWC timer
- Two types of serial interface
 - UART
 - 5, 7, 8 bits transfer data length
 - Serial I/O
- LCD controller/driver
 - 42 segments x 4 commons (max. 168 pixels)
 - Built-in LCD voltage divider

● External interrupts (2 channels)

- Two channels are independent and capable of wake-up from low-power consumption mode (with an edge detection function).

● Standby mode (low-power mode)

- Stop mode (oscillation stops so as to minimize the current consumption).
- Sleep mode (CPU stops so as to reduce the current consumption to approx. 1/3 of normal).

● I/O ports: max. 33 channels

- General-purpose I/O ports (N-ch open-drain): 22 (Also serve as segment pins)
- General-purpose I/O ports (N-ch open-drain): 4 (2 also serve as LCD bias pins)
- General-purpose I/O ports (CMOS): 7 (6 also serve as peripheral pins)

1.2 MB89950/950A Series Product Range

The MB89950/950A series contains 4 different models. Table 1.2-1 "MB89950/950A series product line-up" lists the product range and Table 1.2-2 "Common specifications for the MB89950/950A series" lists the common specifications.

■ MB89950/950A series product range

Table 1.2-1 MB89950/950A series product line-up

	Part number			
	MB89951A	MB89953A	MB89P955	MB89PV950 *2
Classification	Mask ROM		OTP	Piggy-back
ROM size	4K x 8 bits (internal mask ROM)	8K x 8 bits (internal mask ROM)	16K x 8 bits (internal OTP)	32K x 8 bits (external ROM)
RAM size	128 x 8 bits	256 x 8 bits	512 x 8 bits	1024 x 8 bits
Low-power consumption (Standby mode)	Sleep mode and stop mode			
Process	CMOS			
Operating voltage *1	2.7 V to 5.5 V		2.7 V to 6.0 V	

*1: Varies with conditions such as operating frequencies.
*2: Use MBM27C256A as the external ROM

Table 1.2-2 Common specifications for the MB89950/950A series

Parameter		Specification
CPU functions		Number of instructions: 136 Instruction bit length: 8 bits Instruction length: 1 to 3 bytes Data bit length: 1, 8, 16 bits Minimum execution time: 0.80 µs to 12.8 µs at 5 MHz Interrupt processing time: 7.26 µs to 115.2 µs at 5 MHz
Peripheral functions	Ports	General-purpose I/O ports (N-ch open-drain): 22 (also serve as LCD segment pins) *1 General-purpose I/O ports (N-ch open-drain): 4 (two also serve as LCD bias pins) General-purpose I/O ports (CMOS): 7 (6 ports serve as peripherals) Total: 33 (max.)
	20-bit timebase timer	20 bits Interrupt cycle: 6.55 ms, 26.21 ms, 104.86 ms, 419.43 ms at 5 MHz
	Watchdog timer	Reset generate cycle: min. 419.4 ms at 5 MHz
	8-bit PWM timer	8-bit reload timer operation (square wave output; operating frequency: 0.8 µs, 12.8 µs, 51.2 µs at 5 MHz) 8-bit resolution PWM operation (conversion frequency: 204.8 µs - 3.36 s) Event count function
	PWC timer	8-bit interval timer operation 8-bit pulse width measurement (continuous measurement, High-width, Low-width measurement and One-cycle measurement) Operation clock (0.8 µs, 3.2 µs, 25.6 µs at 5 MHz)
	UART	Transfer data length: 5, 7, 8 bits Internal baud rate generator (Max. 78125 bps at 5 MHz)
	8-bit serial I/O	8 bits LSB first/MSB first selectability One clock selectable from four transfer clocks (one external shift clock, three internal shift clocks: 1.6 µs, 6.4 µs, 25.6 µs at 5 MHz)
	LCD controller/ driver	Common output: 4 (max.) Segment output: 42 (max.) Operation mode: 1/2 bias and 1/2 duty, 1/3 bias and 1/3 duty, 1/3 bias and 1/4 duty LCD display RAM size: 21 bytes (42 x 4 bits, max. 168 pixels) Dividing voltage for LCD driving: built-in/external voltage divider selectable
	External interrupt	2 independent channels (interrupt vector, request flag, request output enable) Edge selectability (rising/falling)

Note:

Unless otherwise specified, values given for clock cycle, conversion times, etc. are for 5 MHz operation.

*1: Segment pins can be selected by mask option.

1.3 Differences among Products

This section describes the differences among the 4 products in the MB89950/950A series and lists points to note in product selection.

■ Differences among products and points to note for product selection

Table 1.3-1 Package and corresponding products

Package	Part number			
	MB89951A	MB89953A	MB89P955	MB89PV950
FPT-64P-M09 (LQFP-64, 0.65 mm pitch)	○	○	○	X
MQP-64C-P01 (MQFP-64, 1 mm pitch)	X	X	X	○

○ : Available
X: Not available

● Current consumption

- In the case of the MB89PV950, add the current consumed by the EPROM, which is connected to the top socket.
- When operated at low speed, the product with a one-time PROM (OTPROM) or an EPROM will consume more current than the product with mask ROM. However the current consumption in sleep/ stop mode, is the same.

For more information about the package, see Section 1.6 "Package Dimensions".

- For more information about the current consumption, see the electrical characteristics in the Data Sheet.

● Mask options

Functions that can be selected as options and how to designate these options vary from product to product. Before using, check Appendix C, "Mask Options".

Take particular care on the following points:

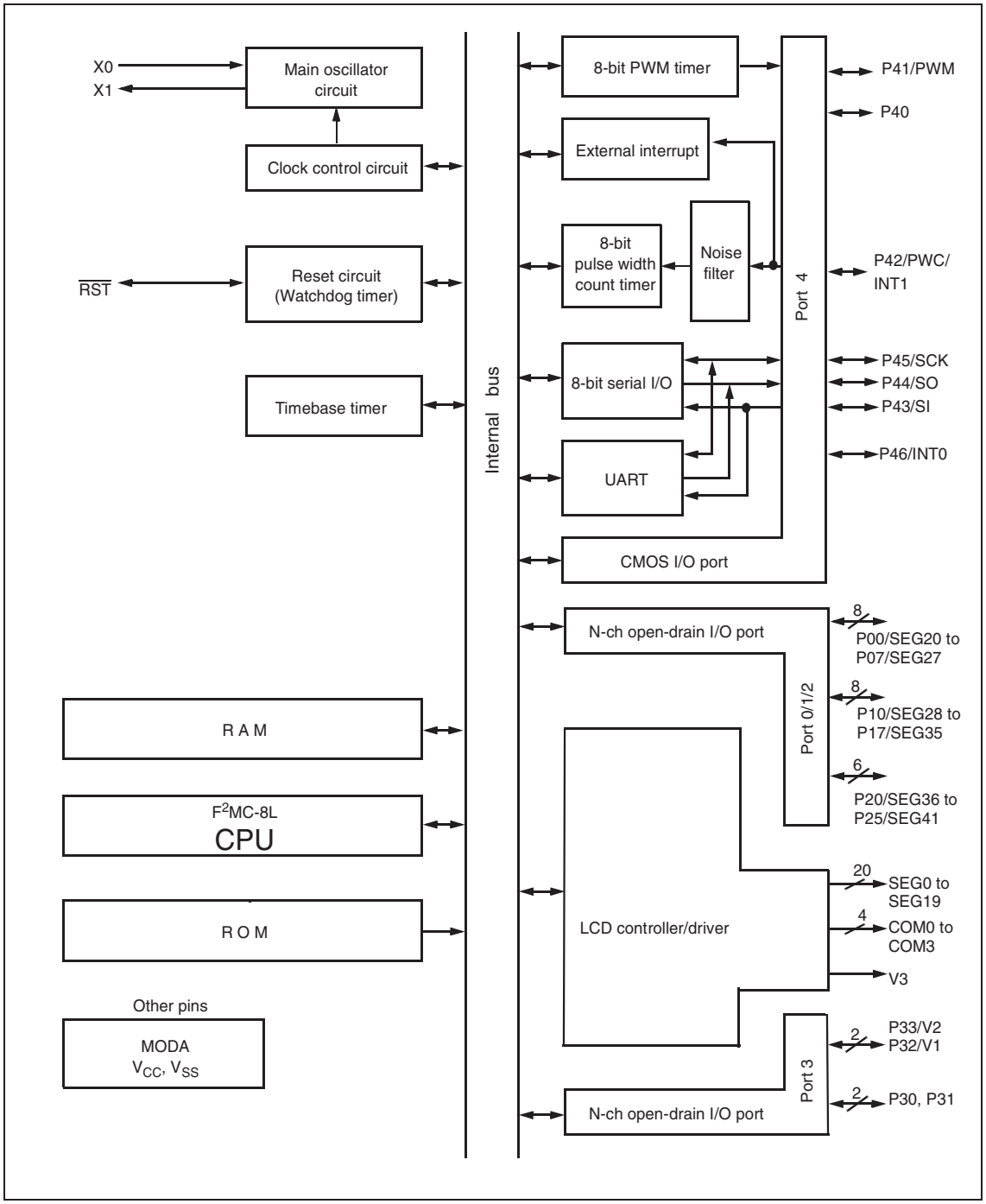
- In the MB89951A and MB89953A, the number of common and segment outputs is specified by Data Release Form.
- Options are fixed on the MB89PV950. (See Appendix C, "Mask Options")

1.4 Block Diagram of MB89950/950A Series

Figure 1.4-1 "MB89950/950A series overall block diagram" shows the block diagram of the MB89950/950A series.

■ MB89950/950A series block diagram

Figure 1.4-1 MB89950/950A series overall block diagram

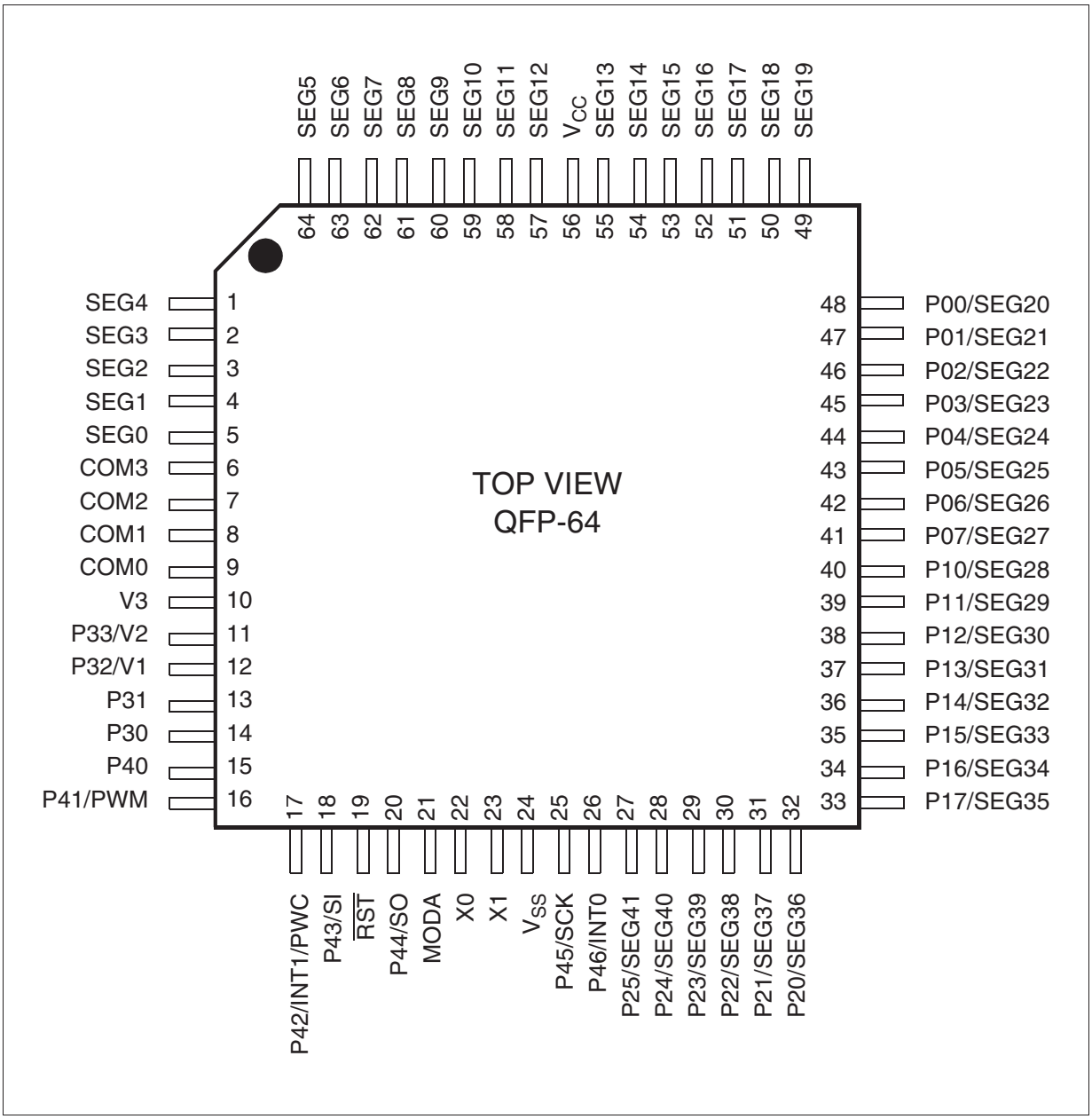


1.5 Pin Assignment

Figure 1.5-1 "FPT-64P-M09 pin assignment" and Figure 1.5-2 "MQP-64C-P01 pin assignment" show the pin assignment diagrams for the MB89950/950A series.

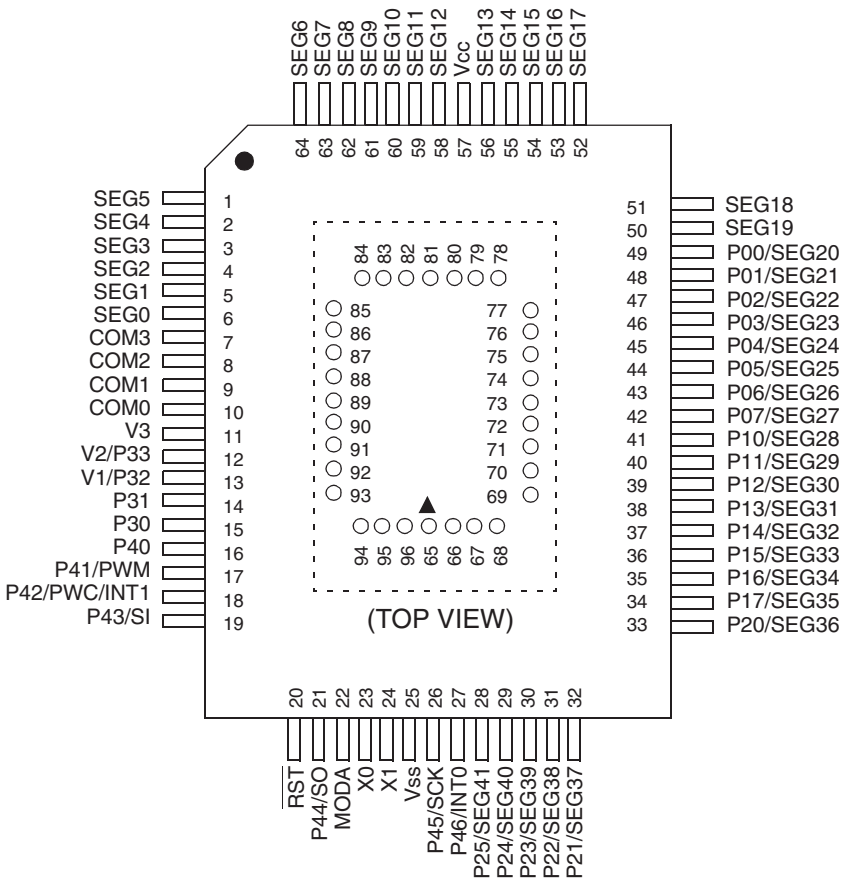
■ FPT-64P-M09 pin assignment

Figure 1.5-1 FPT-64P-M09 pin assignment



■ MQP-64C-P01 pin assignment

Figure 1.5-2 MQP-64C-P01 pin assignment



Pin assignment on package top (MB89PV950 only)

Pin no.	Pin name	Pin no.	Pin name	Pin no.	Pin name	Pin no.	Pin name
65	N.C.	73	A2	81	N.C.	89	$\overline{OE}$
66	Vpp	74	A1	82	O4	90	N.C.
67	A127	5	A0	83	O5	91A	11
68	A7	76	N.C.	84	O6	92	A9
69	A6	77	O1	85	O7	93	A8
70	A5	78	O2	86	O8	94	A13
71	A4	79	O3	87	$\overline{CE}$	95	A14
72	A3	80	Vss	88	A10	96	Vcc

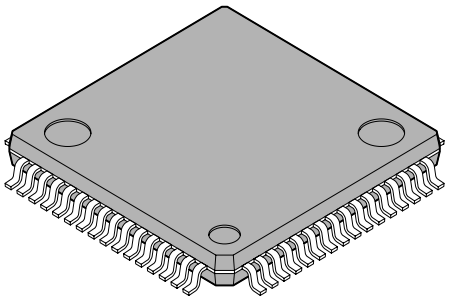
N.C.: Internally connected. Do not use.

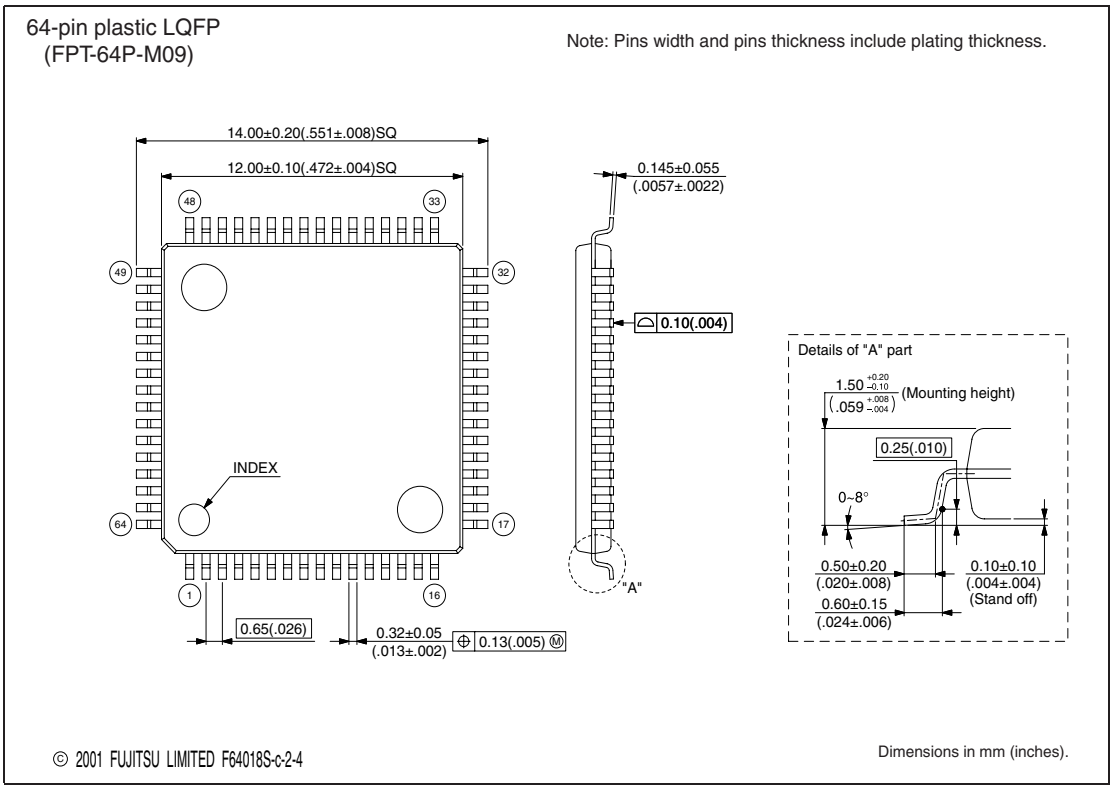
1.6 Package Dimensions

Two types of packages are available for MB89950/950A series. Figure 1.6-1 "FPT-64P-M09 package dimensions" and Figure 1.6-2 "MQP-64C-P01 package dimensions" show the package dimensions.

■ FPT-64P-M09 package dimensions

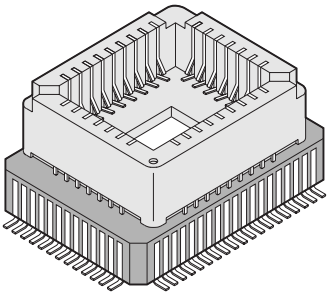
Figure 1.6-1 FPT-64P-M09 package dimensions

64-pin plastic LQFP  (FPT-64P-M09)	Lead pitch	0.65 mm
	Package width × package length	12 × 12 mm
	Lead shape	Gullwing
	Sealing method	Plastic mold
	Mounting height	1.70 mm MAX

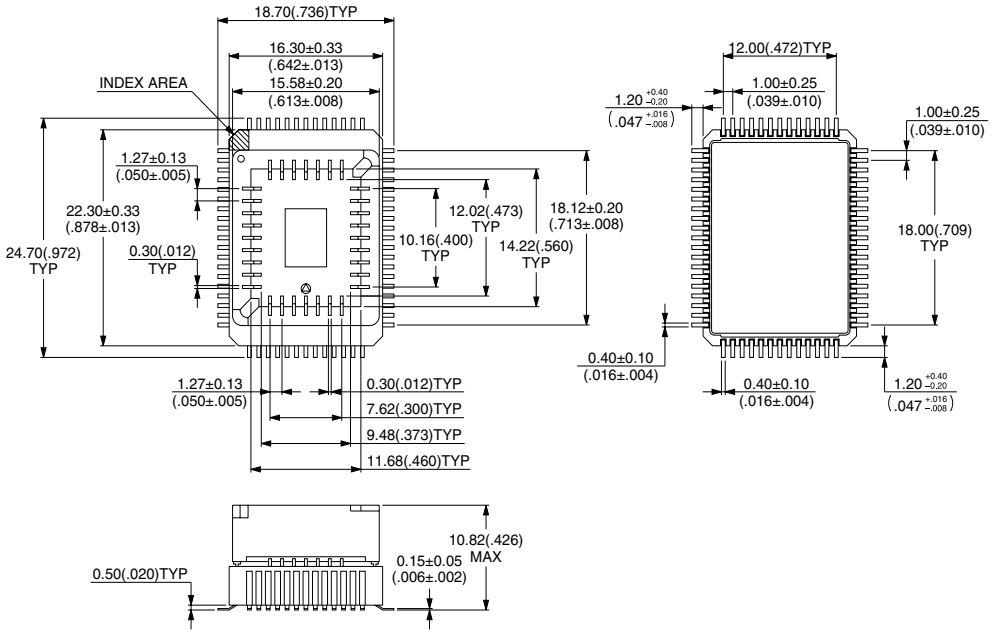


■ MQP-64C-P01 package dimensions

Figure 1.6-2 MQP-64C-P01 package dimensions

64-pin ceramic MQFP  (MQP-64C-P01)	Lead pitch	1.00 mm
	Lead shape	Straight
	Motherboard material	Ceramic
	Mounted package material	Plastic

64-pin ceramic MQFP
(MQP-64C-P01)



© 1994 FUJITSU LIMITED M64004SC-1-3

Dimensions in mm (inches).

1.7 I/O Pins and Pin Functions

Table 1.7-1 "Pin description" and Table 1.7-2 "Pin description for external ROM (for MB89PV950 only)" list the MB89950/950A series I/O pins and their functions. Table 1.7-3 "I/O circuit type" lists the I/O circuit types.
The letter in the "I/O circuit type" column in Table 1.7-1 "Pin description" refer to the letter in the "Type" column Table 1.7-3 "I/O circuit type".

■ I/O pins and pin functions

Table 1.7-1 Pin description (1/2)

Pin no.		Pin name	I/O circuit type	Function
LQFP *1	MQFP *2			
22	23	X0	A	Clock oscillator pins.
23	24	X1		
21	22	MODA	B	Operation mode selection pin. This pin is connected directly to V_{SS} with pull-down resistor.
19	20	$\overline{RST}$	C	Reset I/O pin. This pin consists of an N-ch open-drain output with a pull-up resistor and hysteresis input. A "LOW" level is output from this pin. A "LOW" voltage on this port generates a RESET condition.
48 to 41	49 to 42	P00/SEG20 to P07/SEG27	D	N-channel open-drain type general-purpose I/O ports. Also serve as LCD controller/driver segment outputs. Switching between port output and segment driver output is performed by the mask option.
40 to 33	41 to 34	P10/SEG28 to P17/SEG35	D	N-channel open-drain type general-purpose I/O ports. Also serve as LCD controller/driver segment outputs. Switching between port output and segment driver output is performed by the mask option.
32 to 27	33 to 28	P20/SEG36 to P25/SEG41	D	N-channel open-drain type general-purpose I/O ports. Also serve as LCD controller/driver segment outputs. Switching between port output and segment driver output is performed by the mask option.
14 to 13	15 to 14	P30 to P31	F	N-channel open-drain type general-purpose I/O ports.
12 to 11	13 to 12	P32/V1 to P33/V2	H	N-channel open-drain type general-purpose I/O ports. Also serve as LCD controller/driver power supply.
15	16	P40	E	General-purpose I/O port. A pull-up resistor option is provided.

Table 1.7-1 Pin description (2/2)

Pin no.		Pin name	I/O circuit type	Function
LQFP *1	MQFP *2			
16	17	P41/PWM	E	General-purpose I/O port. Also serves as PWM timer toggle output (PWM). A pull-up resistor option is provided.
17	18	P42/PWC/INT1	E	General-purpose I/O port. Also serves as pulse-width count timer input (PWC) and external interrupt input (INT1). The PWC and INT1 inputs are hysteresis type. A pull-up resistor option is provided.
18	19	P43/SI	E	General-purpose I/O port. Also serves as serial I/O and UART data input (SI). The SI input is hysteresis type. A pull-up resistor option is provided.
20	21	P44/SO	E	General-purpose I/O port. Also serves as serial I/O and UART data output (SO). A pull-up resistor option is provided.
25	26	P45/SCK	E	General-purpose I/O port. Also serves as serial I/O and UART clock input/output (SCK). The SCK input is hysteresis type. A pull-up resistor option is provided.
26	27	P46/INT0	E	General-purpose input port. Also serves as external interrupt input (INT0). The input is hysteresis type. A pull-up resistor option is provided.
5 to 1 64 to 57 55 to 49	6 to 1 64 to 58 56 to 50	SEG0 to SEG19	G	For LCD segment driver outputs.
9 to 6	10 to 7	COM0 to COM3	G	For LCD common driver outputs.
10	11	V3	-	For LCD driver power supply.
56	57	V_{CC}	-	Power pin.
24	25	V_{SS}	-	Power (GND) pin.

*1: FPT-64P-M09
*2: MQP-64C-P01

Table 1.7-2 Pin description for external ROM (for MB89PV950 only)

Pin no.	Pin name	I/O	Function
66	V_{PP}	O	For high-level output.
67	A12	O	For address output.
68	A7		
69	A6		
70	A5		
71	A4		
72	A3		
73	A2		
74	A1		
75	A0		
77	O1	I	For data input.
78	O2		
79	O3		
80	V_{SS}	O	For power supply (GND).
82	O4	I	For data input.
83	O5		
84	O6		
85	O7		
86	O8		
87	CE	O	For ROM chip enable. The High level is output in standby mode.
88	A10	O	For address output.
89	OE	O	For ROM output enable. The Low level is always output.
91	A11	O	For address output.
92	A9		
93	A8		
94	A13	O	For address output.
95	A14		
96	V_{CC}	O	For EPROM power supply.
65	N.C.	--	For internal connection. Keep open.
76			
81			
90			

Table 1.7-3 I/O circuit type (1/2)

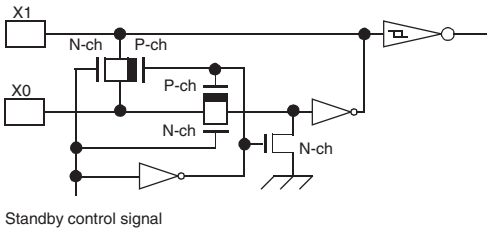
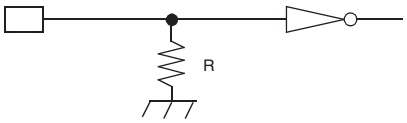
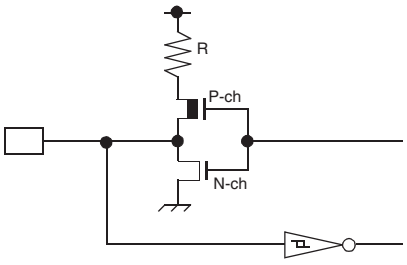
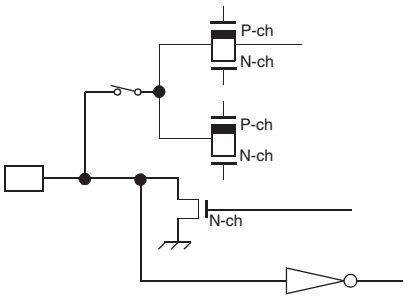
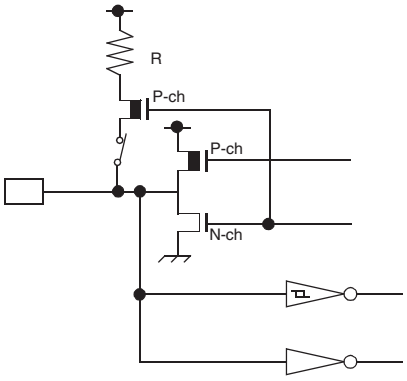
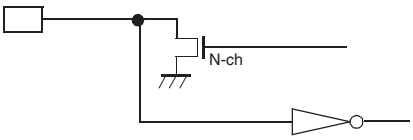
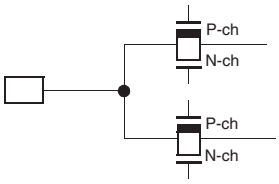
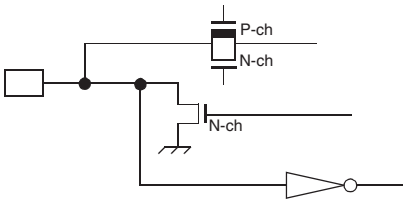
Type	Circuit	Remarks
A		• Crystal oscillator • Feedback resistor: About 1 MΩ (5 V)
B		• CMOS input • Pull-down resistor (N-ch): About 50 kΩ (5 V)
C		• Output pull-up resistor (P-ch): About 50 kΩ (5 V) • Hysteresis input
D		• N-ch open-drain output • CMOS input • The segment driver output is optional.

Table 1.7-3 I/O circuit type (2/2)

Type	Circuit	Remarks
E		• CMOS output • CMOS input • Hysteresis input (peripheral input) • The pull-up resistor is optional: About 50 kΩ (5 V)
F		• N-ch open-drain output • CMOS input
G		• LCD controller/driver common/segment driver output
H		• N-ch open-drain output • CMOS input

CHAPTER 2
HANDLING DEVICES

This chapter describes points to note when using the general-purpose single-chip microcontroller.

2.1 "Notes on Handling Devices"

2.1 Notes on Handling Devices

This section lists points to note regarding the power supply voltage, pins, and other device handling aspects.

■ Notes on handling devices

● Preventing latch-up

Latch-up may occur on CMOS ICs if voltage higher than V_{CC} or lower than V_{SS} is applied to input and output pins other than medium to high-voltage pins or if higher than the voltage which shows on Absolute Maximum Ratings is applied between V_{CC} and V_{SS}.

When latch-up occurs, supply current increases rapidly and might thermally damage elements. Take great care not to exceed the absolute maximum ratings in circuit operation.

● Treatment of unused input pins

Leaving unused input pins open could cause malfunctions. They should be connected to pull-up or pull-down resistor.

● Power supply voltage fluctuations

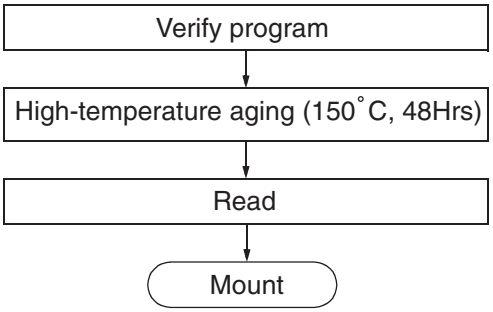
Although V_{CC} power supply voltage is assured to operate within the rated, a rapid change to the IC is therefore cause malfunctions, even if it occurs within the rated range. Stabilizing voltage supplied of the IC is therefore important. As stabilization guidelines, it is recommended to control power so that V_{CC} ripple fluctuations (P-P. value) will be less that 10% of the standard V_{CC} value at the commercial frequency (50 to 60 Hz) and the transient fluctuation rate will be less than 0.1 V/ms at the time of a momentary fluctuation such as when power is switched.

● Precaution when using an external clock

Even when an external clock is used, oscillation stabilization time is required for power-on reset (option selection) and release from stop mode.

● Recommended screening conditions

The OTPROM product should be screened by high-temperature aging before mounting.



The programming test cannot be performed for all bits of the preprogrammed OTPROM product due to its characteristics. Consequently, 100% programming yielding cannot be ensured.

● Treatment of N.C. pins

Be sure to leave (internally connected) N.C. pins open.

● Unused LCD controller/driver dedicated pins

When LCD controller/driver dedicated pins are not in use, keep it open.

● Port shared with SEG pin

When using port shared with SEG pin, be sure that the input voltage to port does not exceed the voltage of V3 (SEG driving voltage). When power-on or reset, SEG pin will output an initial value of "L".

● LCD controller/driver not in use

When LCD controller/driver is not in use, connect the V3 pin to V_{CC} and keep other LCD controller/driver dedicated pins open.

CHAPTER 3

CPU

This chapter describes the functions and operation of the CPU.

3.1 "Memory Space"

3.2 "Dedicated Registers"

3.3 "General-purpose Registers"

3.4 "Interrupts"

3.5 "Resets"

3.6 "Clocks"

3.7 "Standby Modes (Low-power Consumption)"

3.8 "Memory Access Mode"

3.1 Memory Space

The microcontrollers of the MB89950/950A series offer a memory space of 64 Kbytes. The memory space contains the I/O area, RAM area, ROM area, and external area. The memory space contains areas used for special purposes such as the general-purpose registers and vector table.

■ Memory space structure

● I/O area (addresses: 0000_H to $007F_H$)

- Control registers and data registers for the internal peripheral functions are located in this area.
- As the I/O area is allocated within the memory space, I/O can be accessed in the same way as memory. High-speed access using direct addressing is available.

● RAM area

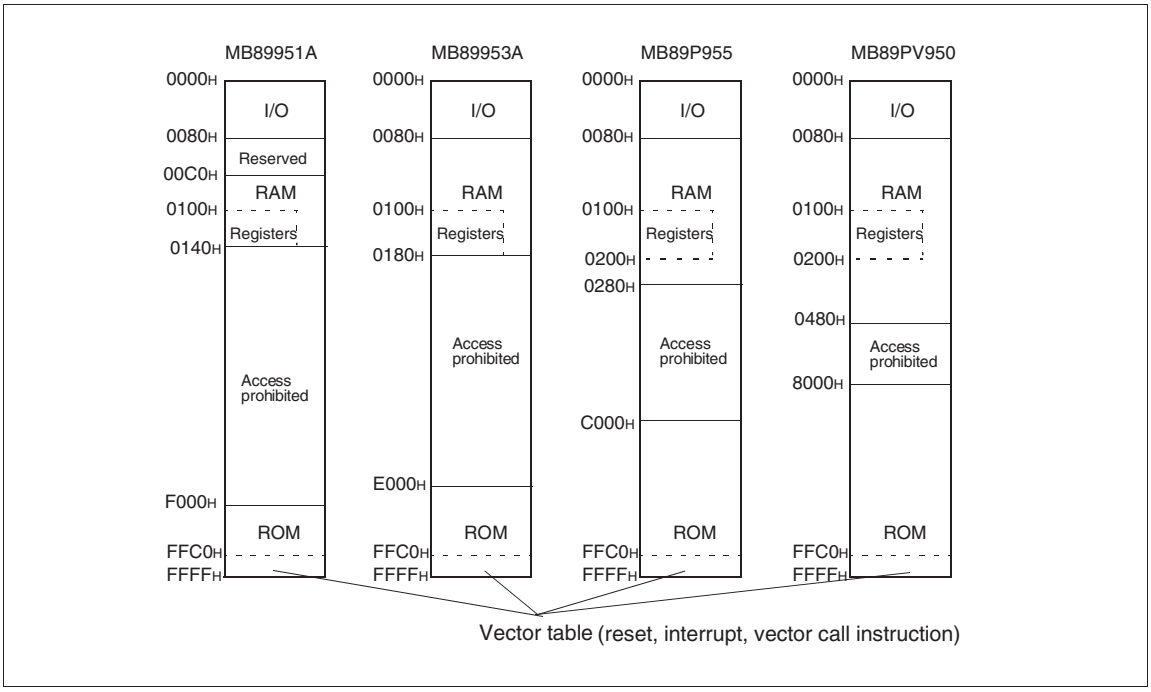
- Internal static RAM is provided as an internal data area.
- The internal RAM size differs from product to product.
- Addresses between 0080_H and $00FF_H$ support high-speed access using direct addressing.
- Addresses between 0100_H and $01FF_H$ can be used as the general-purpose register area (restrictions apply for some products).
- The contents of RAM is indeterminate after a reset.

● ROM area

- Internal ROM is provided as an internal program area.
- The internal ROM size differs from product to product.
- Addresses between $FFC0_H$ and $FFFF_H$ are used for the vector table, etc.

■ Memory map

Figure 3.1-1 Memory map



3.1.1 Special Areas

In addition to the I/O area, the special purpose areas in the memory space include the general-purpose register area and the vector table area.

■ General-purpose register areas (addresses: 0100_H to $01FF_H$)

- Provides auxiliary registers for 8-bit arithmetic operation and transfer instructions.
- Allocated to a region of the RAM area. Can also be used as normal RAM.
- Using the area as general-purpose registers enables high-speed access by general-purpose register addressing using short instructions.

Table 3.1-1 "General-purpose register areas" lists the areas in each device that can be used for general-purpose registers.

Table 3.1-1 General-purpose register areas

Part number	MB89951A	MB89953A	MB89P955/PV950
Number of banks	8	16	32
Address range	0100_H to $013F_H$	0100_H to $017F_H$	0100_H to $01FF_H$

See section 3.2.2 "Register Bank Pointer (RP)" and section 3.3 "General-purpose Registers" for details.

■ Vector table area (addresses: $FFC0_H$ to $FFFF_H$)

- Used as the vector table for the vector call instruction, interrupts, and resets.
- The vector table is allocated at the top of the ROM area. The start address of the corresponding processing routine is set as data at each vector table address.

Table 3.1-2 "Vector table" lists the vector table addresses referenced by the vector call instruction, interrupts, and resets.

See Section 3.4 "Interrupts", Section 3.5 "Resets", and "(6) CALLV #vct" in Appendix B.2, "Special Instructions" for details.

Table 3.1-2 Vector table

Vector call instruction	Vector table address	
	Upper	Lower
CALLV #0	$FFC0_H$	$FFC1_H$
CALLV #1	$FFC2_H$	$FFC3_H$
CALLV #2	$FFC4_H$	$FFC5_H$
CALLV #3	$FFC6_H$	$FFC7_H$
CALLV #4	$FFC8_H$	$FFC9_H$
CALLV #5	$FFCA_H$	$FFCB_H$
CALLV #6	$FFCC_H$	$FFCD_H$
CALLV #7	$FFCE_H$	$FFCF_H$

Interrupts	Vector table address	
	Upper	Lower
IRQB	$FFE4_H$	$FFE5_H$
IRQA	$FFE6_H$	$FFE7_H$
IRQ9	$FFE8_H$	$FFE9_H$
IRQ8	$FFEA_H$	$FFEB_H$
IRQ7	$FFEC_H$	$FFED_H$
IRQ6	$FFEE_H$	$FFEF_H$
IRQ5	$FFF0_H$	$FFF1_H$
IRQ4	$FFF2_H$	FFF3H
IRQ3	$FFF4_H$	$FFF5_H$
IRQ2	$FFF6_H$	$FFF7_H$
IRQ1	$FFF8_H$	$FFF9_H$
IRQ0	$FFFA_H$	$FFFB_H$
Mode data	-- (*1)	$FFFD_H$
Reset vector	$FFFE_H$	$FFFF_H$

*1: $FFFC_H$ is not available. (Set FF_H.)

3.1.2 Storing 16-bit Data in Memory

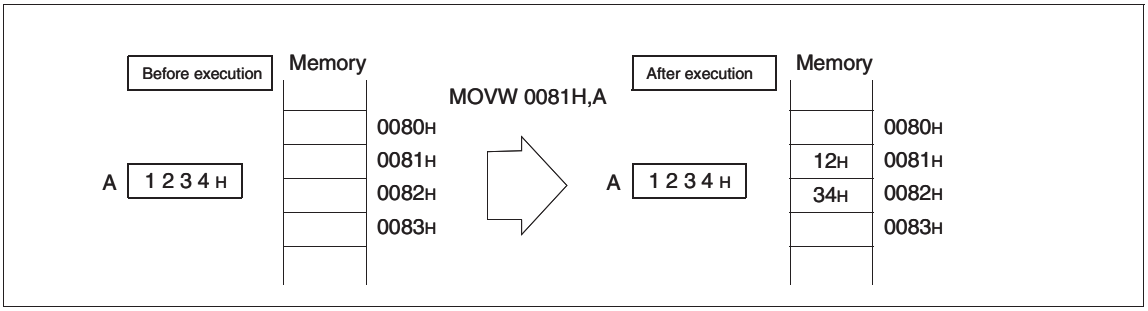
For 16-bit data and the stack, store the upper data in the lower memory address value.

■ Storing 16-bit data in RAM

When writing 16-bit data to memory, store the upper byte at the lower address and the lower byte at the next address. Handle reading of 16-bit data in the same way.

Figure 3.1-2 "Storing 16-bit data in memory" shows how 16-bit data is stored in memory.

Figure 3.1-2 Storing 16-bit data in memory



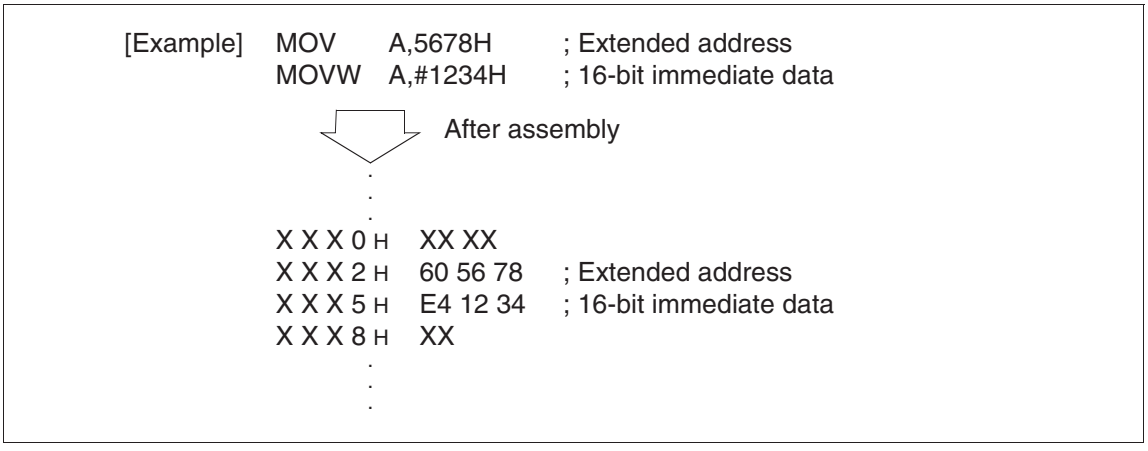
■ Storing 16-bit operands

The same byte order applies when specifying a 16-bit operand in an instruction. Store the upper byte at the address following the operation code (instruction) and the lower byte at the next address.

The byte ordering applies to both 16-bit immediate data and operands that specify a memory address.

Figure 3.1-3 "Byte order of 16-bit data in an instruction" shows how 16-bit data is stored in an instruction.

Figure 3.1-3 Byte order of 16-bit data in an instruction



■ Storing 16-bit data on stack

The same byte order applies when saving 16-bit register data on the stack during an interrupt or similar. The upper byte is stored in the lower address.

3.2 Dedicated Registers

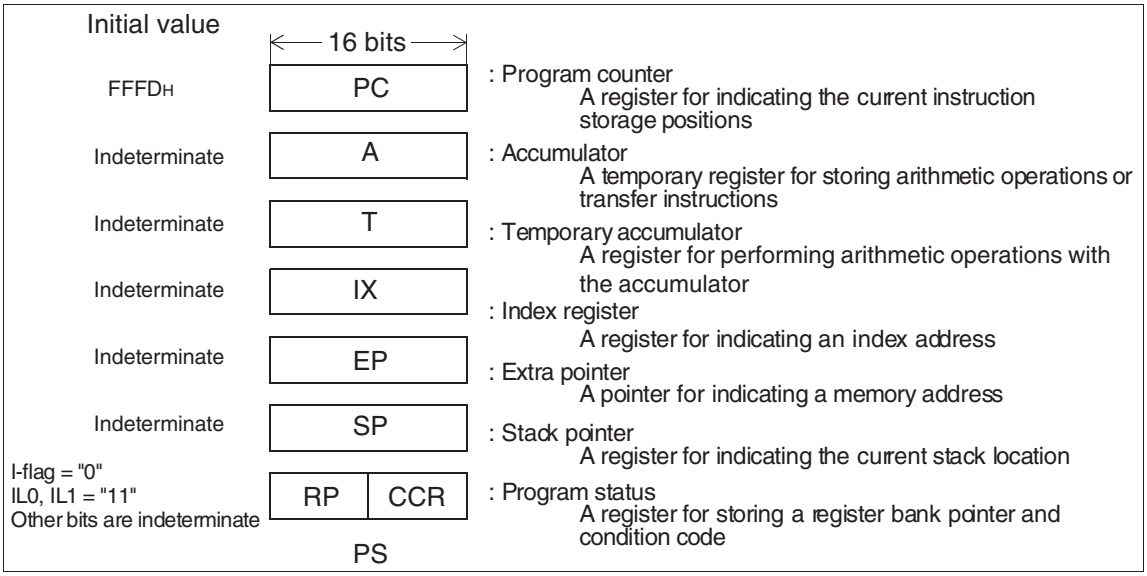
The dedicated registers in the CPU consist of the program counter (PC), two arithmetic operation registers (A and T), three address pointers (IX, EP, and SP), and the program status (PS). All registers are 16 bits.

■ Dedicated register configuration

The dedicated registers in the CPU consist of seven 16-bit registers. Some of these registers are also able to be used as 8-bit register, using the lower 8 bits only.

Figure 3.2-1 "Dedicated register configuration" shows the structure of the dedicated registers.

Figure 3.2-1 Dedicated register configuration



■ Dedicated register functions

● Program counter (PC)

The program counter is a 16-bit counter that indicates the memory address of the instruction currently being executed by the CPU. Instruction execution, interrupts, resets, and similar update the contents of the program counter. The initial value during a reset is the read address of the mode data ($FFFD_H$).

● Accumulator (A)

The accumulator is a 16-bit arithmetic operation register. The accumulator is used to perform arithmetic operations and data transfers with data in memory or in other registers such as the temporary accumulator (T). The content of the accumulator can be treated as either word (16-bit) or byte (8-bit) data. Only the lower 8 bits (AL) of the accumulator are used for byte arithmetic operations or transfers. In this case, the upper 8 bits (AH) remain unchanged. The content of the accumulator after a reset is indeterminate.

● Temporary accumulator (T)

The temporary accumulator is an auxiliary 16-bit arithmetic operation register used to perform arithmetic operations with the data in the accumulator (A). The content of the temporary accumulator is treated as word data (16-bit) for word-length arithmetic operations with the accumulator and as byte data (8-bit) for byte-length arithmetic operations. For byte-length arithmetic operations, only the lower 8 bits of the temporary accumulator (TL) are used and the upper 8 bits (TH) are not used.

Executing a transfer instruction to transfer data to the accumulator (A) automatically transfer the previous content of the accumulator to the temporary accumulator. In this case also, a byte transfer leaves the upper 8 bits of the temporary accumulator (TH) unchanged. The content of the temporary accumulator after a reset is indeterminate.

● Index register (IX)

The index register is a 16-bit register used to hold the index address. The index register is used in conjunction with a single byte offset value (-128 to +127). Adding the sign-extended offset value to the index address generates the memory address for data access. The content of the index register after a reset is indeterminate.

● Extra pointer (EP)

The extra pointer is a 16-bit register used to hold a memory address for data access. The content of the extra pointer after a reset is indeterminate.

● Stack pointer (SP)

The stack pointer is a 16-bit register used to hold the address referenced during operations such as interrupts, subroutine calls, and the stack save and restore instructions. The value of the stack pointer during program execution is the address of the most recently saved data on the stack. The content of the stack pointer after a reset is indeterminate.

● Program status (PS)

The program status is a 16-bit control register. The upper 8 bits contain the register bank pointer (RP) which points to the address of the current general-purpose register bank.

The lower 8 bits contain the condition code register (CCR) which contains flags indicating the current CPU status. The two 8-bit registers which form the program status cannot be accessed independently (the program status can only be accessed by the MOVW A,PS and MOVW PS,A instructions).

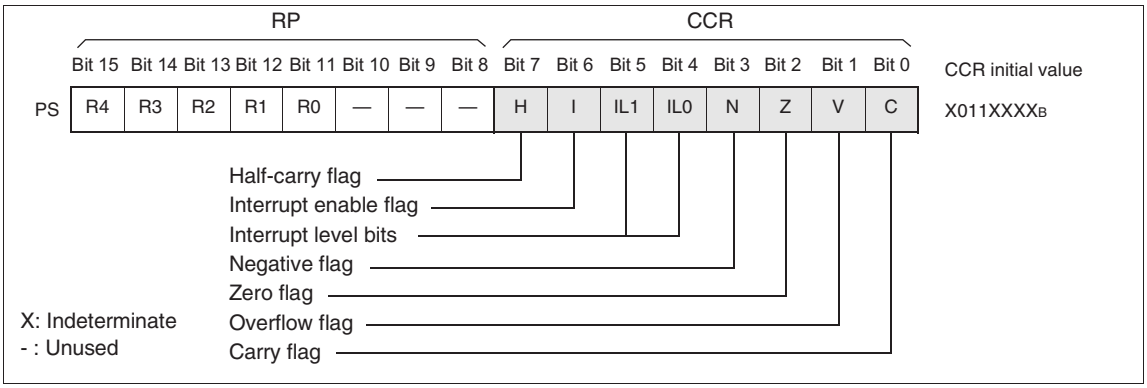
Refer to the F^2MC-8L MB89600 series Programming Manual for details on using the dedicated registers.

3.2.1 Condition Code Register (CCR)

The condition code register (CCR) located in the lower 8 bits of the program status (PS) consists of the C, V, Z, N, and H bits indicating the results of arithmetic operations and the contents of transfer data, and the I, IL1, and IL0 bits for control whether or not the CPU accepts interrupt requests.

■ Structure of condition code register (CCR)

Figure 3.2-2 Structure of condition code register



■ Arithmetic operation result bits

● Half-carry flag (H)

Set to "1" when a carry from bit 3 to bit 4 or a borrow from bit 4 to bit 3 occurs as a result of an arithmetic operation. Clear to "0" otherwise. As this flag is for the decimal adjustment instructions, do not use this flag in cases other than addition or subtraction.

● Negative flag (N)

Set to "1" if the most significant bit (MSB) is set to "1" as a result of an arithmetic operation. Clear to "0" when the bit is set to "0".

● Zero flag (Z)

Set to "1" when an arithmetic operation results in "0". Clear to "0" otherwise.

● Overflow flag (V)

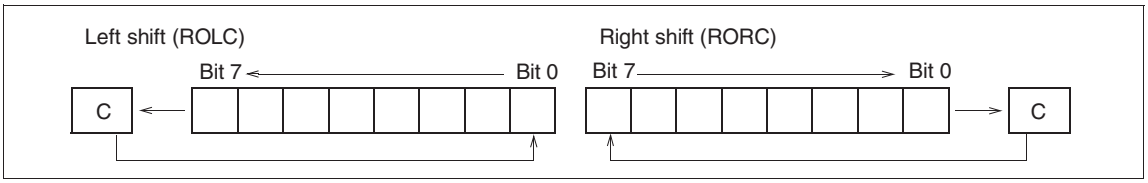
Set to "1" if a signed numeric value overflows because of an arithmetic calculation. Clear to "0" if the overflow does not occur.

● Carry flag (C)

Set to "1" when a carry from bit 7 or borrow to bit 7 occurs as a result of an arithmetic operation. Clear to "0" otherwise. Set to the shift-out value in case of a shift instruction.

Figure 3.2-3 "Change of carry flag by shift instruction" shows the change of the carry flag by a shift instruction.

Figure 3.2-3 Change of carry flag by shift instruction



Note:

The condition code register is part of the program status (PS) and cannot be accessed independently.

Reference:

In practice, the flag bits are rarely fetched and used directly. Instead, the bits are used indirectly by instructions such as branch instructions (such as BNZ) or the decimal adjustment instructions (DAA, DAS). The content of the flags after a reset is indeterminate.

■ Interrupt acceptance control bit

● Interrupt enable flag (I)

Interrupt is enabled when this flag is set to "1" and the CPU accepts interrupt. Interrupt is prohibited when this flag is set to "0" and the CPU does not accept interrupt.

The initial value after a reset is "0".

Normal practice is to set the flag to "1" by the SETI instruction and clear to "0" by the CLRI instruction.

● Interrupt level bits (IL1, IL0)

These bits indicate the level of the interrupt currently being accepted by the CPU. The value is compared with the interrupt level setting registers (ILR1 to ILR3) which have a setting for each peripheral function interrupt request (IRQ0 to IRQB).

Given that the interrupt enable flag is enabled (I = "1"), the CPU only performs interrupt processing for interrupt requests with an interrupt level value that is less than the value of these bits. Table 3.2-1 "Interrupt level" lists the interrupt level priorities. The initial value after a reset is "11_B".

Table 3.2-1 Interrupt level

IL1	IL0	Interrupt level	Priority
0	0	1	High ↑
0	1		
1	0	2	
1	1	3	↓ Low (no interrupt)

Reference:

The interrupt level bits (IL1, IL0) are normally "11_B" when the CPU is not processing an interrupt (during main program execution).

See Section 3.4 "Interrupts" for details on interrupts.

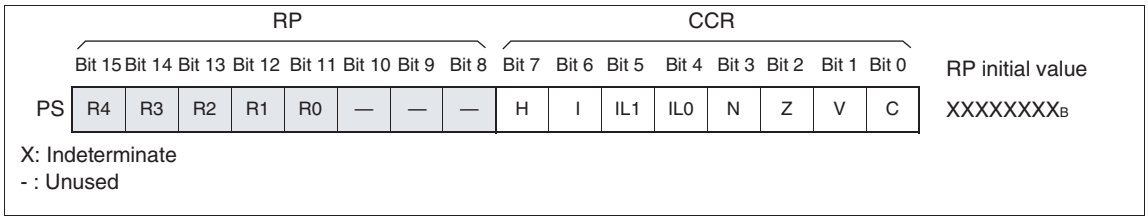
3.2.2 Register Bank Pointer (RP)

The register bank pointer (RP) located in the upper 8 bits of the program status (PS) indicates the address of the general-purpose register bank currently in use. The RP is converted to form the actual address in general-purpose register addressing.

■ Structure of register bank pointer (RP)

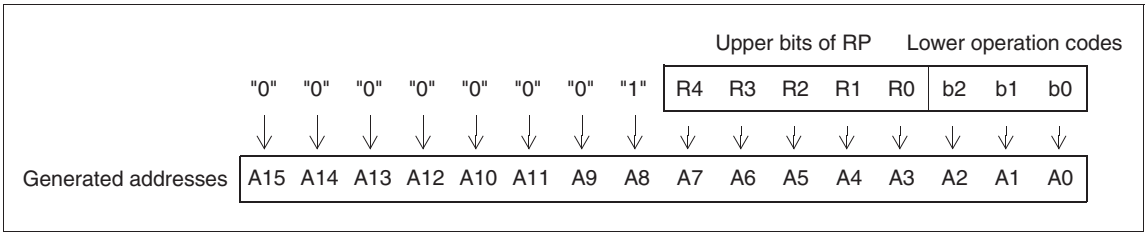
Figure 3.2-4 "Structure of register bank pointer" shows the structure of the register bank pointer.

Figure 3.2-4 Structure of register bank pointer



The register bank pointer indicates the address of the register bank currently in use. Figure 3.2-5 "Rule for conversion of actual addresses of general-purpose register area" shows the relationship between the pointer contents and the actual address is based on the conversion rule.

Figure 3.2-5 Rule for conversion of actual addresses of general-purpose register area



The register bank pointer points to the memory block (register bank) in the RAM area that is used for general-purpose registers. A total of 32 register banks are available. A register bank is specified by setting a value between 0 and 31 in the upper 5 bits of the register bank pointer. Each register bank contains eight 8-bit general-purpose registers. Registers are specified by the lower 3 bits of the operation codes.

Using the register bank pointer, the addresses 0100_H to $01FF_H$ can be used as the general-purpose register area. However, the available area is limited on some products if internal RAM only is used. The initial value after a reset is indeterminate.

Note:

The register bank pointer is part of the program status (PS) and cannot be accessed independently.

3.3 General-purpose Registers

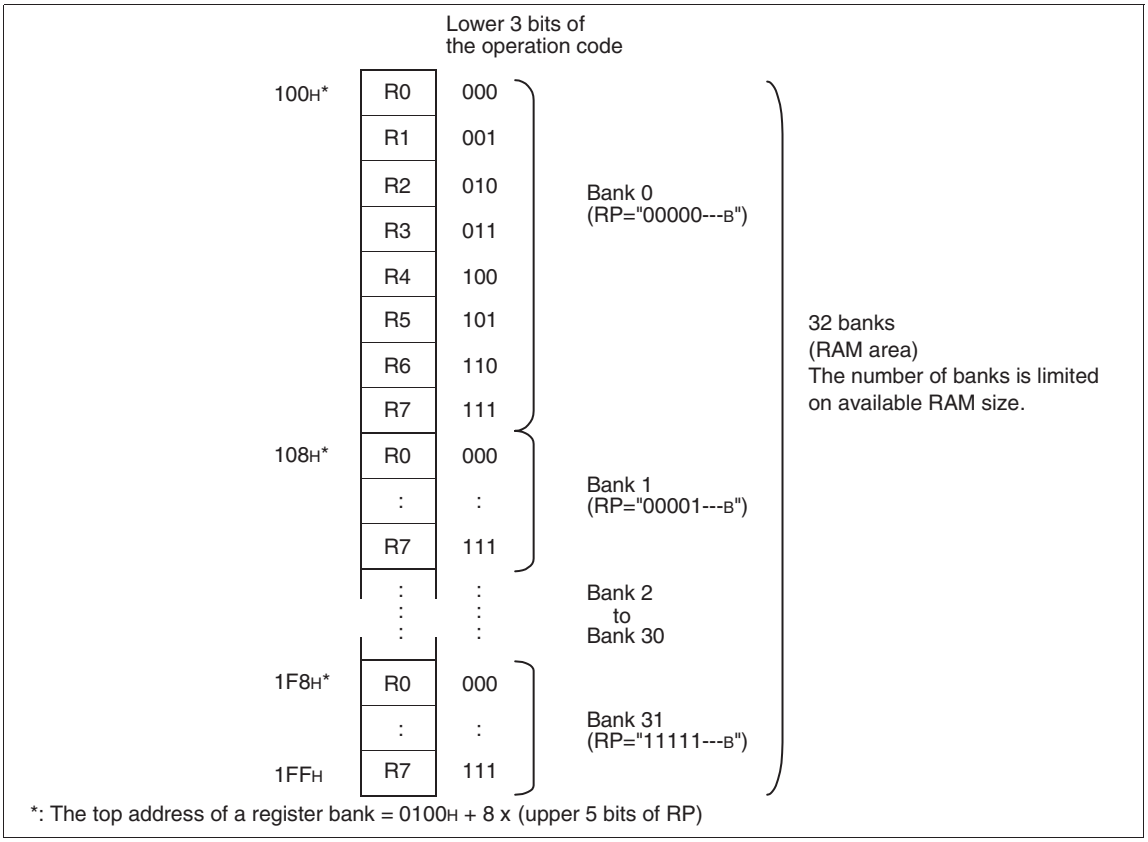
The general-purpose registers are a memory block made up of banks, with 8 x 8-bit registers per bank.
The register bank pointer (RP) is used to specify the register bank.
The function permits the use of up to 32 banks, but the number of banks that can actually be used depends on how much RAM the device has.
Register banks are valid for interrupt processing, vector call processing, and subroutine calls.

■ Structure of general-purpose registers

- The general-purpose registers are 8 bits and located in the register banks of the general-purpose register area (in RAM).
- One bank contains eight registers (R0 to R7) and up to a total of 32 banks. However, the number of banks available for general-purpose registers is limited on some products if internal RAM only is used.
- The register bank currently in use is specified by the register bank pointer (RP). The lower three bits of the operation code specify general-purpose register 0 (R0) to general-purpose register 7 (R7).

Figure 3.3-1 "Register bank structure" shows the register bank structure.

Figure 3.3-1 Register bank structure



See Section 3.1.1 "Special Areas" for the general-purpose register area available for each product.

■ Features of general-purpose registers

General-purpose registers have the following features:

- RAM can be accessed at high-speed using short instructions (general-purpose register addressing).
- Registers are grouped in blocks in the form of register banks. This simplifies the process of saving register contents and dividing registers by function.

Dedicated register banks can be permanently assigned for each interrupt processing or vector call (CALLV #0 to #7) processing routine by general-purpose register. For example, register bank 4 interrupt 2.

For example, a particular interrupt processing routine only uses a particular register bank which cannot be written to unintentionally by other routines. The interrupt processing routine only needs to specify its dedicated register bank at the start of the routine to effectively save the general-purpose registers in use prior to the interrupt. Therefore, saving the general-purpose registers to the stack or other memory location is not necessary. This allows high-speed interrupt handling while maintaining simplicity.

Also, as an alternative to saving general-purpose registers in subroutine calls, register banks can be used to create reentrant programs (programs that do not use fixed addresses and can be entered more than once) usually made by the index register (IX).

Note:

If an interrupt processing routine changes the register bank pointer (RP), ensure that the program does not also change the interrupt level bits in the condition code register (CCR: IL1, IL0) when specifying the register bank.

3.4 Interrupts

The MB89950/950A series has 12 interrupt request inputs corresponding to peripheral functions. The interrupt level can be set independently.
If an interrupt request output is enabled in the peripheral function, an interrupt request from a peripheral function is compared with the interrupt level in the interrupt controller. The CPU performs interrupt operation according to how the interrupt is accepted. The CPU wakes up from standby mode, and returns to the interrupt or normal operation.

■ Interrupt requests from peripheral functions

Table 3.4-1 "Interrupt request and interrupt vector" lists the interrupt requests corresponding to the peripheral functions. On acceptance of an interrupt, execution branches to the interrupt processing routine. The contents of interrupt the vector table address corresponding to the interrupt request specifies the branch destination address for the interrupt processing routine.

An interrupt processing level can be for each interrupt request in the interrupt level setting registers (ILR1, ILR2, ILR3). Three levels are available.

If an interrupt request with the same or lower level occurs during execution of an interrupt processing routine, the latter interrupt is not normally processed until the current interrupt processing routine completes. If interrupt request set the same level occur simultaneously, the highest priority is IRQ0.

Table 3.4-1 Interrupt request and interrupt vector

Interrupt request	Vector table address		Bit names of the interrupt level setting register	Priority (*1)
	Upper	Lower		
IRQ0 (External interrupt 0)	$FFFA_H$	$FFFB_H$	L01, L00	High ↑ Low
IRQ1 (External interrupt 1)	$FFF8_H$	$FFF9_H$	L11, L10	
IRQ2 (8-bit PWM timer)	$FFF6_H$	$FFF7_H$	L21, L20	
IRQ3 (PWC)	$FFF4_H$	$FFF5_H$	L31, L30	
IRQ4 (UART)	$FFF2_H$	$FFF3_H$	L41, L40	
IRQ5 (8-bit serial I/O)	$FFF0_H$	$FFF1_H$	L51, L50	
IRQ6 (Timebase timer)	$FFEE_H$	$FFEF_H$	L61, L60	
IRQ7 (Unused)	$FFEC_H$	$FFED_H$	L71, L70	
IRQ8 (Unused)	$FFEA_H$	$FFEB_H$	L81, L80	
IRQ9 (Unused)	$FFE8_H$	$FFE9_H$	L91, L90	
IRQA (Unused)	$FFE6_H$	$FFE7_H$	LA1,LA0	
IRQB (Unused)	$FFE4_H$	$FFE5_H$	LB1, LB0	

*1: This priority is applied when interrupts of the same level occur simultaneously.

3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3)

The interrupt level setting registers (ILR1, ILR2, ILR3) together contain 12 blocks of 2-bit data, with each data corresponding to an interrupt request from a peripheral function. The interrupt level for each interrupt is set in that interrupt's corresponding 2-bit data (interrupt level setting bits).

■ Structure of interrupt level setting registers (ILR1, ILR2, ILR3)

Figure 3.4-1 Structure of interrupt level setting registers

Register	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
ILR1	007CH	L31	L30	L21	L20	L11	L10	L01	L00	11111111B
		W	W	W	W	W	W	W	W	
ILR2	007DH	L71	L70	L61	L60	L51	L50	L41	L40	11111111B
		W	W	W	W	W	W	W	W	
ILR3	007EH	LB1	LB0	LA1	LA0	L91	L90	L81	L80	11111111B
		W	W	W	W	W	W	W	W	

W: Write-only

Two bits of the interrupt level setting registers are allocated to each interrupt request. The value of the interrupt level setting bits in these registers sets the interrupt priority (interrupt levels 1 to 3).

The interrupt level setting bits are compared with the interrupt level bits in the condition code register (CCR: IL1, IL0).

The CPU does not accept interrupt requests set to interrupt level 3.

Table 3.4-2 "Interrupt level setting bit and interrupt level" shows the relationship between the interrupt level setting bits and the interrupt levels.

Table 3.4-2 Interrupt level setting bit and interrupt level

L01 to LB1	L00 to LB0	Interrupt request level	Priority
0	0	1	High ↕ Low (no interrupt)
0	1		
1	0	2	
1	1	3	

Reference:

The interrupt level bits in the condition code register (CCR: IL1, IL0) are normally "11_B" during main program execution.

Note:

As the IRL1, ILR2, and ILR3 registers are write-only, the bit manipulation instructions cannot be used.

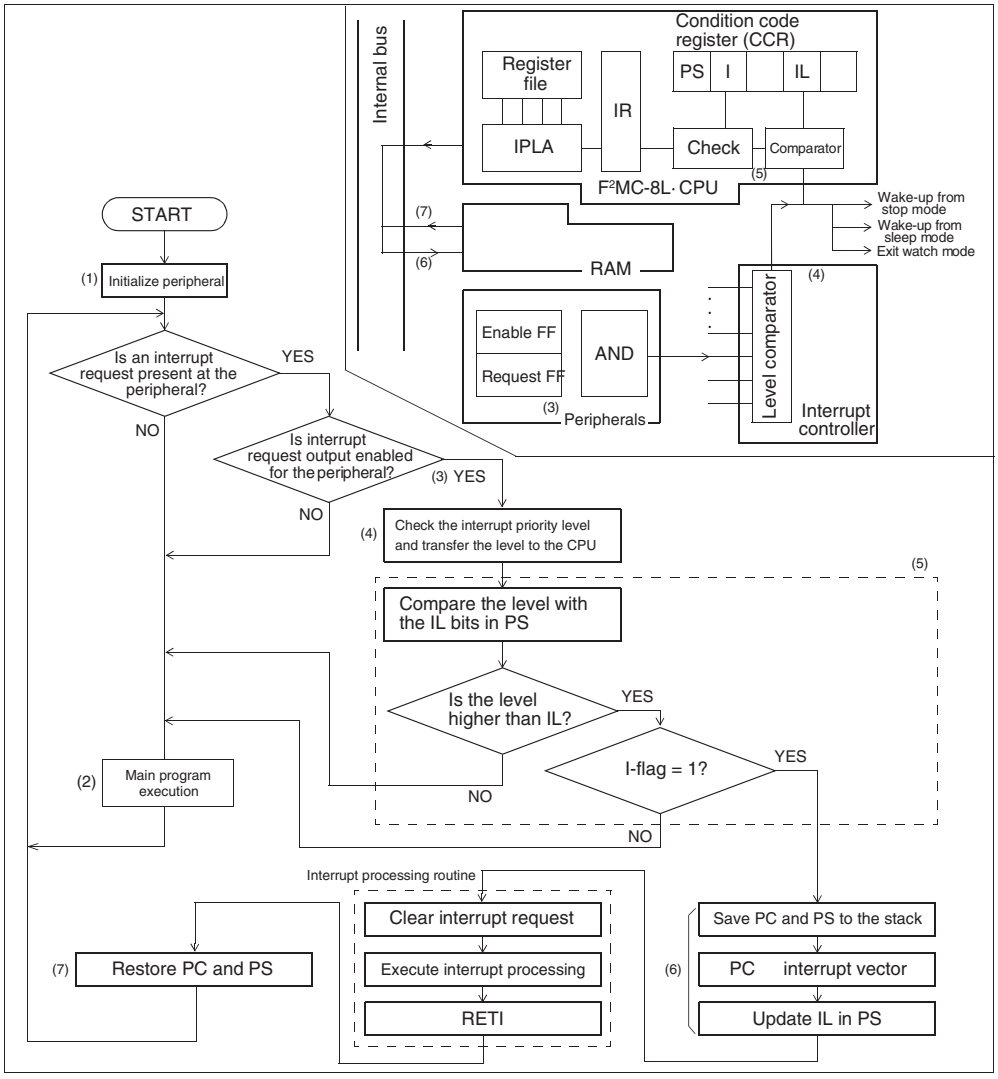
3.4.2 Interrupt Processing

The interrupt controller transmits the interrupt level to the CPU when an interrupt request is generated by a peripheral function. If the CPU is able to receive the interrupt, the CPU temporarily halts the currently executing program and executes the interrupt processing routine.

■ Interrupt processing

The procedure for interrupt operation is performed in the following order: interrupt source generated at peripheral function, set the interrupt request flag bit (request FF), discriminate the interrupt request enable bit (enable FF), the interrupt level (ILR1, ILR2, ILR3 and CCR: IL1, IL0), simultaneously generated interrupt requests with the same level, then check the interrupt enable flag (CCR: I). Figure 3.4-2 "Interrupt processing" shows the interrupt processing.

Figure 3.4-2 Interrupt processing



1. After a reset, all interrupt requests are disabled.
 - Initialize the peripheral functions that are to generate interrupts in the peripheral function initialization program, set the interrupt levels in the appropriate interrupt level setting registers (ILR1, ILR2, ILR3), and start peripheral function.
 - The interrupt level can be set to 1, 2 or 3. Level 1 is the highest priority, followed by level 2. Setting level 3 disables the interrupt for that peripheral function.
2. Execute the main program (for multiple interrupts, execute the interrupt processing routine).
3. The interrupt request flag bit (request FF) for a peripheral function is set to "1" when the peripheral function generates an interrupt source. If the interrupt request enable bit for the peripheral function is set to "enable" (enable FF = "1"), the peripheral function outputs the interrupt request to the interrupt controller.
4. The interrupt controller continuously monitors for interrupt requests from the peripheral functions and passes the interrupt level of the current interrupt request with the highest interrupt level to the CPU. The interrupt controller also evaluates the priority order if requests with the same level are present simultaneously.
5. If the interrupt level received by the CPU has a higher priority (a lower level value) than the level set in the interrupt level bits in the condition code register (CCR: IL1, IL0), the CPU checks the interrupt enable flag (CCR: I) and receives the interrupt if interrupts are enabled (CCR: I = "1").
6. The CPU saves the contents of the program counter (PC) and program status (PS) on the stack, reads the top address of the interrupt processing routine from the interrupt vector table for the interrupt, updates the interrupt level bits in the condition code register (CCR: IL1, IL0) with the received interrupt level, and starts execution of the interrupt processing routine.
7. Finally, on execution of the RETI instruction, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution from the instruction following the last instruction executed before the interrupt.

Note:

As the interrupt request flag bit of a peripheral function is not cleared automatically when an interrupt request is received, the bit must be cleared by the program (normally, by writing "0" to the interrupt request flag bit) at interrupt processing routine.

An interrupt wakes up the CPU from standby mode (low-power consumption). See Section 3.7 "Standby Modes (Low-power Consumption)" for details.

Reference:

If the interrupt request flag bit is cleared at the top of the interrupt processing routine, the peripheral function that has generated the interrupt becomes able to generate another interrupt during execution of the interrupt processing routine (resetting the interrupt request flag bit). However, the interrupts are not normally accepted until the current processing routine completes.

3.4.3 Multiple Interrupts

Multiple interrupts can be performed by setting different interrupt levels to the interrupt level setting register for two or more interrupt requests from peripheral functions.

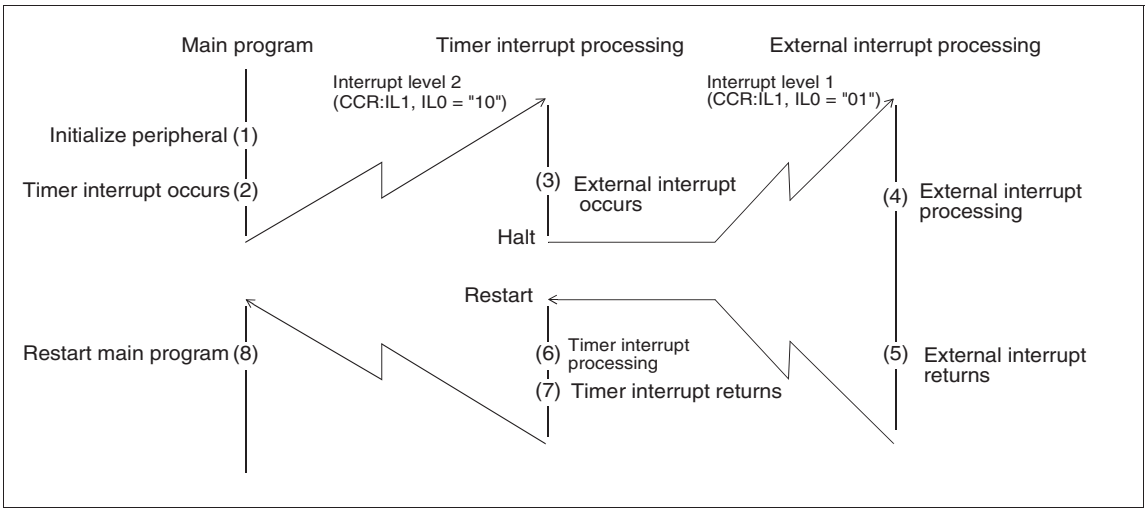
■ Multiple interrupts

If the interrupt request having the higher interrupt levels occurs during the interrupt processing routines, the CPU halts the current interrupt process and switches to accept the interrupt with the higher priority. Interrupt levels can be set in the range 1 to 3. However, the CPU does not accept interrupt requests set to interrupt level 3.

● Example of multiple interrupts

As an example of multiple interrupt processing, assume that an external interrupt has a higher priority than the timer interrupt. The timer interrupt is set to level 2 and the external interrupt is set to level 1. Figure 3.4-3 "Example of multiple interrupts" shows the processing when the external interrupt occurs during execution of timer interrupt processing.

Figure 3.4-3 Example of multiple interrupts



- During execution of timer interrupt processing, the interrupt level bits in the condition code register (CCR:IL1, IL0) are automatically set to the same value as the interrupt level setting register (ILR1, ILR2, ILR3) corresponding to the timer interrupt (level 2 in this example). If the interrupt request set to higher interrupt level (level 1 in this example) occurs at this time, the interrupt processing has priority.
- To temporarily disable multiple interrupts during the timer interrupt, the interrupt enable flag in the condition code register is set to "interrupts disabled" (CCR: I = "0") or the interrupt level bits (IL1, IL0) set to "00_B".
- On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution of the interrupted program.
 Restoring the program status (PS) returns the condition code register (CCR) to the value prior to the interrupt.

3.4.4 Interrupt Processing Time

The total time from the generation of an interrupt request until control passes to the interrupt processing routine is the sum of the time required to complete execution of the current instruction and the interrupt handling time (the time required to prepare for interrupt processing). The maximum time for this process is 30 instruction cycles.

■ Interrupt processing time

When an interrupt request occurs, the time until the interrupt is accepted and the interrupt processing routine is executed includes the interrupt request sampling time and the interrupt handling time.

● Interrupt request sampling time

Whether or not an interrupt request has occurred is determined by sampling and testing for interrupt requests during the final cycle of each instruction. Therefore, the CPU is unable to identify interrupt requests during execution of an instruction. The longest delay occurs when an interrupt request is generated immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles).

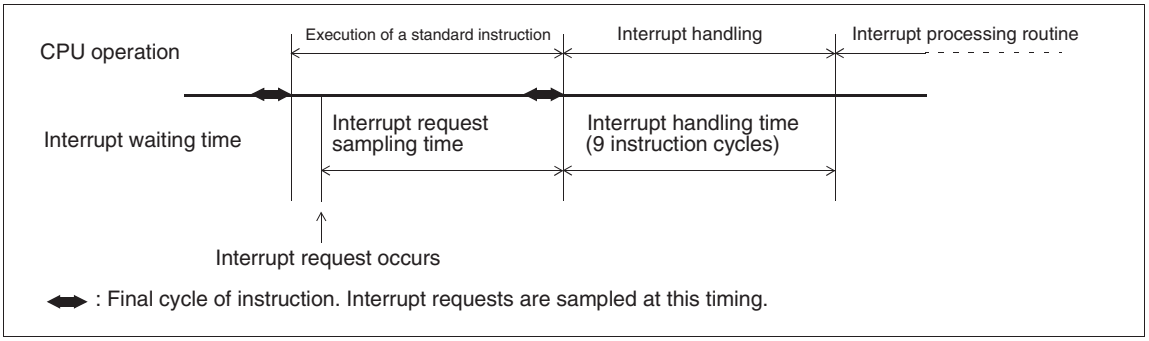
● Interrupt handling time

Nine instruction cycles are required to perform the following preparation for interrupt processing after the CPU accepts an interrupt request:

- Save the program counter (PC) and program status (PS).
- Set the top address of the interrupt processing routine (the interrupt vector) in the PC.
- Update the interrupt level bits (PS: CCR: IL1, IL0) in the program status (PS).

Figure 3.4-4 "Interrupt processing time" shows the interrupt processing time.

Figure 3.4-4 Interrupt processing time



The total interrupt processing time of 21 + 9 = 30 instruction cycles is required if an interrupt request occurs immediately after starting execution of a DIVU instruction, which has the longest instruction cycles (21 instruction cycles). If, on the other hand, the program does not use the DIVU or MULU instructions, the maximum interrupt processing time is 6 + 9 = 15 instruction cycles.

The time of one instruction cycle changes with the clock mode and the main clock frequency as selected by the "speed-shift" (gear) function. See Section 3.6 "Clocks" for details.

3.4.5 Stack Operation during Interrupt Processing

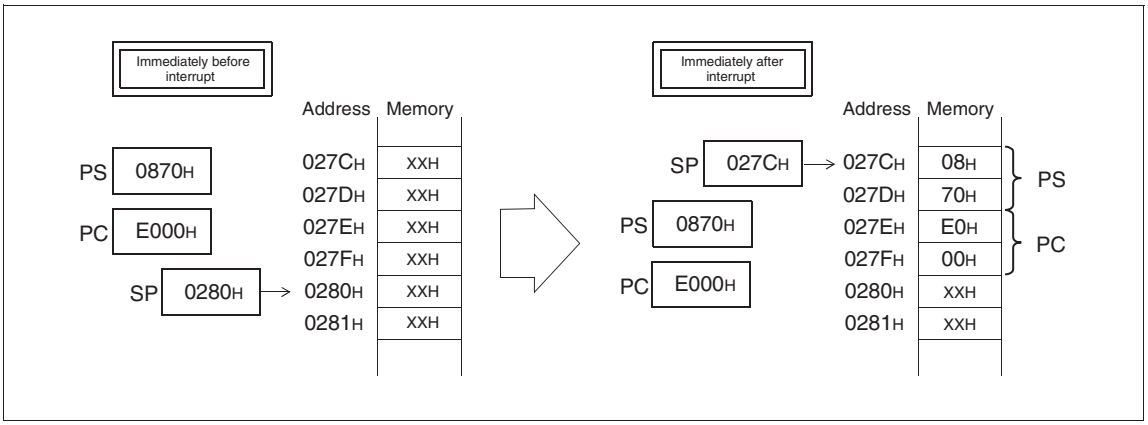
This section describes the saving of the register contents to the stack and restore operation during interrupt processing.

■ Stack operation at start of interrupt processing

The CPU automatically saves the current contents of the program counter (PC) and program status (PS) to the stack when an interrupt is accepted.

Figure 3.4-5 "Stack operation at start of interrupt processing" shows the stack operation at the start of interrupt processing.

Figure 3.4-5 Stack operation at start of interrupt processing



■ Stack operation at interrupt return

On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU performs the opposite processing to interrupt initiation, restoring first the program status (PS) and then the program counter (PC) from the stack. This returns the PS and PC to their states immediately prior to the start of the interrupt.

Note:

The CPU does not automatically save the accumulator (A) or temporary accumulator (T) contents to the stack. Use the PUSHW and POPW instructions to save and restore A and T contents to and from the stack.

3.4.6 Stack Area for Interrupt Processing

Interrupt processing execution uses the stack area in RAM. The contents of the stack pointer (SP) specifies the top address of the stack area.

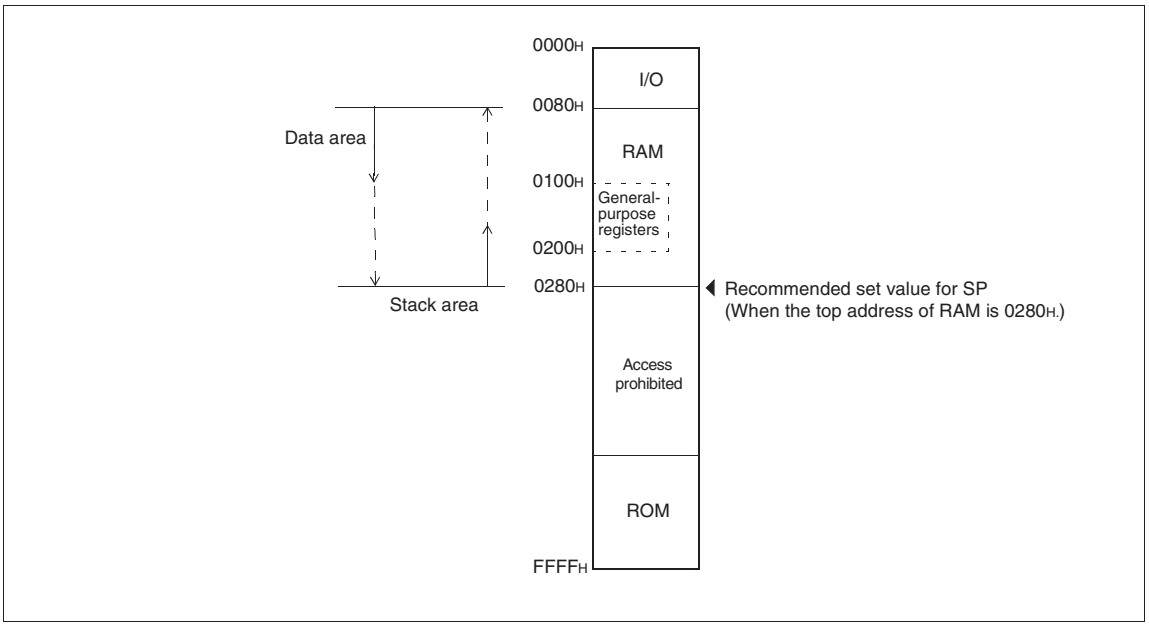
■ Stack area for interrupt processing

The subroutine call instruction (CALL) and vector call instruction (CALLV) use the stack area to save and restore the program counter (PC). The stack area is also used by the PUSHW and POPW instructions to temporarily save and restore registers.

- The stack area is located in RAM along with the data area.
- Initializing the stack pointer (SP) to the top address of RAM and allocating data areas upwards from the bottom RAM address is recommended.

Figure 3.4-6 "Stack area for interrupt processing" shows the example of stack area setting.

Figure 3.4-6 Stack area for interrupt processing



Note:

The stack area is used in the downward direction starting from a high address by functions such as interrupts, subroutine calls, and the PUSHW instruction. Instructions such as return instructions (RETI, RET) and the POPW instruction release stack area in the upward direction. Take care when the stack address is decreased by multiple interrupts or subroutine calls that the stack does not overlap the general-purpose register area or areas containing other data.

3.5 Resets

The MB89950/950A series supports the following four types of reset source:

- **External reset**
- **Software reset**
- **Watchdog reset**
- **Power-on reset (optional)**

At reset, main clock oscillation stabilization delay time may or may not occur by the operating mode and option settings.

■ Reset source

Table 3.5-1 Reset source

Reset source	Reset condition
External reset	Set the external reset pin to the "L" level.
Software reset	Write "0" to the software reset bit in the standby control register (STBC: RST).
Watchdog reset	Watchdog timer overflow.
Power-on reset	Power is turned on (only on products with a power-on reset).

● External reset

Inputting an "L" level to the external reset pin ($\overline{RST}$) generates an external reset. Returning the reset pin to the "H" level wakes up the CPU from the external reset.

When power is turned on to products with power-on reset or for external resets in stop mode, the reset operation is performed after the oscillation stabilization delay time has passed and the CPU wakes up from the external reset. External resets on products without power-on reset do not wait for the oscillation stabilization delay time.

The external reset pin can also function as a reset output pin (optional).

● Software reset

Writing "0" to the software reset bit in the standby control register (STBC: RST) generates a four-instruction-cycle reset. The software reset does not wait for the oscillation stabilization delay time.

● Watchdog reset

The watchdog reset generates a four-instruction-cycle reset if data is not written to the watchdog timer control register (WDTC) within a fixed time after the watchdog timer starts. The watchdog reset does not wait for the oscillation stabilization delay time.

● Power-on reset

Products can be set to with or without power-on reset (optional). On products with power-on reset, turning on the power generates a reset. The reset operation is performed after the oscillation stabilization delay time has passed. Moreover, external reset signal is outputted by the reset output option.

On products without power-on reset, an external reset circuit is required to generate a reset when the power is turned on.

■ Main clock oscillation stabilization delay time and the reset source

Whether there will be an oscillation stabilization delay time depends on the operating mode when reset occurs, and the power-on reset option selected.

Following reset, operation always starts out in the normal main clock operating mode, regardless of the kind of reset it was, or the operating mode (the clock mode and standby mode) prior to reset. Therefore, if reset occurs while the main clock oscillator is stopped or in a stabilization delay time, the system will be in a "main clock oscillation stabilization reset" state, and a clock stabilization period will be provided. If the device is set for no power-on reset, however, no main clock oscillation stabilization delay time is provided for power-on or external reset.

In software or watchdog reset, if the reset occurs while the device is in main clock mode, no stabilization time is provided.

Table 3.5-2 "Reset source and oscillation stabilization delay time" shows the relationships between the reset sources and the main clock oscillation stabilization delay time, and reset mode (mode fetch) operations.

Table 3.5-2 Reset source and oscillation stabilization delay time

Reset source	Operating state	Reset operation and main clock oscillation stabilization delay time	
		With power-on reset	Without power-on reset
External reset (*1)	At power-on, during stop mode	After the main clock oscillation stabilization delay time, if the external reset is waked up, reset is operated. (*2)	Reset state is held until external reset is waked up; then the reset is operated.
Software and watchdog reset	Main clock mode	After 4-instruction-cycle reset occurs, reset is operated. (*3)	
Power-on reset		Device enters main clock oscillation stabilization delay time at power-on. Reset is operated after delay time ends. (*2)	An external circuit must be provided to hold external reset asserted at power-on until main clock has had time to stabilize.

*1: No oscillation stabilization delay time is required for external reset while main clock mode is operating. Reset is operated after external reset is waked up.
*2: If the reset output option is selected, "L" is output at $\overline{\text{RST}}$ pin during the main clock oscillation stabilization delay time.
*3: If the reset output option is selected, "L" level is output at $\overline{\text{RST}}$ pin during 4-instruction-cycle.

3.5.1 External Reset Pin

Inputting an "L" level to the external reset pin generates a reset. If products are set to with the reset output (optional), the pin outputs an "L" level depending on internal reset sources.

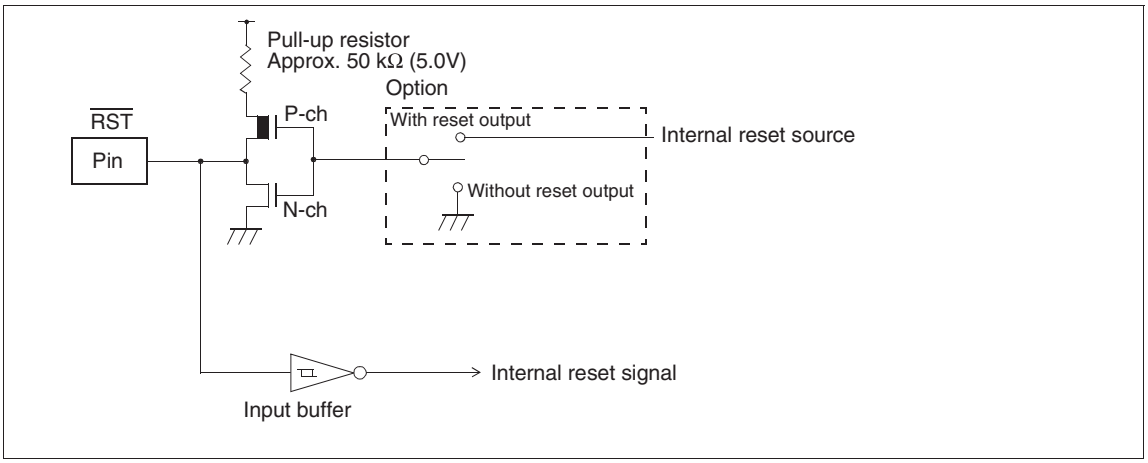
■ Block diagram of external reset pin

The external reset pin ($\overline{RST}$) on products with the reset output is a hysteresis input type and N-ch open-drain output type with a pull-up resistor.

The external reset pin on products without a reset output option is only for the reset input.

Figure 3.5-1 "Block diagram of external reset pin" shows the block diagram of the external reset pin.

Figure 3.5-1 Block diagram of external reset pin



■ External reset pin functions

Inputting an "L" level to the external reset pin ($\overline{RST}$) generates an internal reset signal.

When selecting products with reset output (option setting), the pin outputs an "L" level depending on internal reset sources or during the oscillation stabilization delay time due to an external reset. Software reset, watchdog reset, and power-on reset are classed as internal reset sources.

Note:

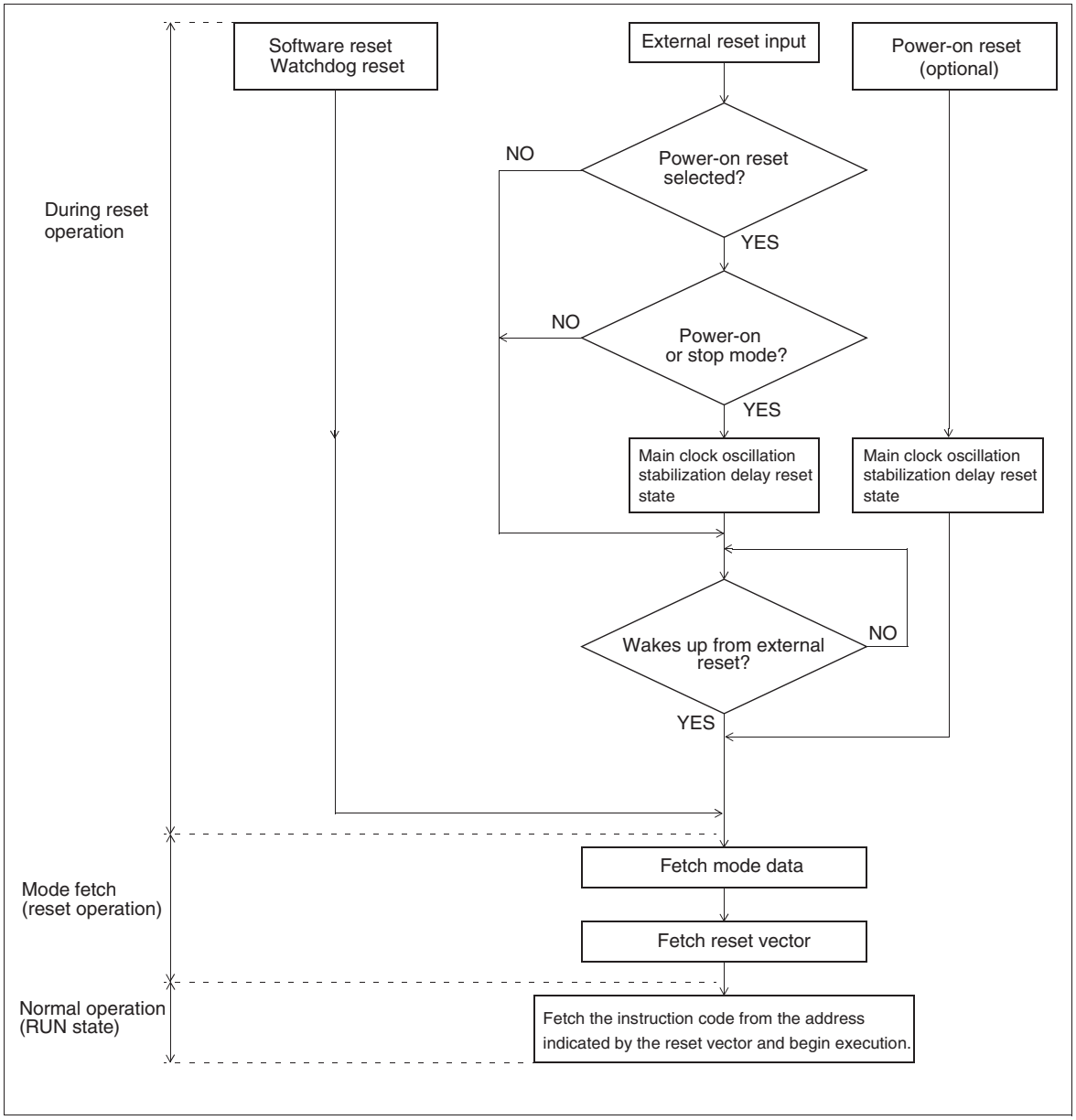
The external reset input accepts asynchronous with the internal clock. Therefore, initialization of the internal circuit requires a clock. Especially when an external clock is used, a clock is needed to be input at the reset.

3.5.2 Reset Operation

When the CPU wakes up from a reset, the CPU selects the read address of the mode data and reset vector according to the mode pin settings, then performs a mode fetch. The mode fetch is performed after the oscillation stabilization delay time has passed when power is turned on to a product with power-on reset, or on wake-up from stop mode by a reset. If reset occurs during a write to RAM, the contents of the RAM address cannot be assured.

■ Overview of reset operation

Figure 3.5-2 Reset operation flow diagram



■ Mode pin

The MB89950/950A series devices are single-chip mode devices. The mode pin (MODA) must be tied to V_{SS}. The mode pin settings determine whether the mode data and reset vector are read from internal ROM.

Do not change the mode pin settings, even after the reset has completed.

■ Mode fetch

When the CPU wakes up from a reset, the CPU reads the mode data and reset vector from internal ROM.

● Mode data (address: $FFFD_H$)

Always set the mode to "00_H" (single-chip mode).

● Reset vector (address: $FFFE_H$ (upper), $FFFF_H$ (lower))

Contains the address where execution is to start after completion of the reset. The CPU starts executing instructions from the address contained in the reset vector.

■ Oscillation stabilization delay reset state

On products with power-on reset, the reset operation for a power-on reset or external reset in stop (main clock) mode starts after the main clock oscillation stabilization delay time selected by the stabilization delay time option. If the CPU has not woken up from the external reset input when the delay time completes, the reset operation does not start until the CPU wakes up from external reset.

As the oscillation stabilization delay time is also required when an external clock is used, a reset requires that the external clock is input.

The main clock oscillation stabilization delay time is timed by the timebase timer.

On products without power-on reset, the oscillation stabilization delay reset state is not used. Therefore, for such products, hold the external reset pin ($\overline{RST}$) at the "L" level to disable the CPU operation until the source oscillation stabilizes.

■ Effect of reset on RAM contents

The contents of RAM are unchanged before and after a reset other than power-on reset. If an external reset is input close to a write timing, however, the contents of the write address cannot be assured. For this reason, all RAM locations being used should be initialized following reset.

3.5.3 Pin States during Reset

Reset initializes the pin states.

■ Pin states during reset

When a reset source occurs, with a few exceptions, all I/O pins (peripheral pins) go to the high-impedance state and the mode data is read from internal ROM (pins with a pull-up resistor (optional) go to the "H" level).

■ Pin states after reading mode data

With a few exceptions, the I/O pins remain in the high-impedance state immediately after reading the mode data (pins with a pull-up resistor (optional) go to the "H" level).

Note:

For devices connected to pins that change to high-impedance state when a reset source occurs take care that malfunction does not occur due to the change in the pin states.

See Appendix E "MB89950/950A Series Pin States" for pin states at the time other than reset.

3.6 Clocks

The clock generator provides an internal oscillation circuit. By connecting with external resonator, the circuits generate the high speed main clock sources. Alternatively, externally generated clock input can be used.
Clock controller controls the speed and supply of the clock signal according to the standby mode.

■ Clock supply map

Oscillation of a clock and its supply to the CPU and peripheral circuit (peripheral functions) are controlled by the clock controller. As shown in the map, operating clocks fed to the CPU and peripheral circuits are affected by standby (sleep/stop) mode.

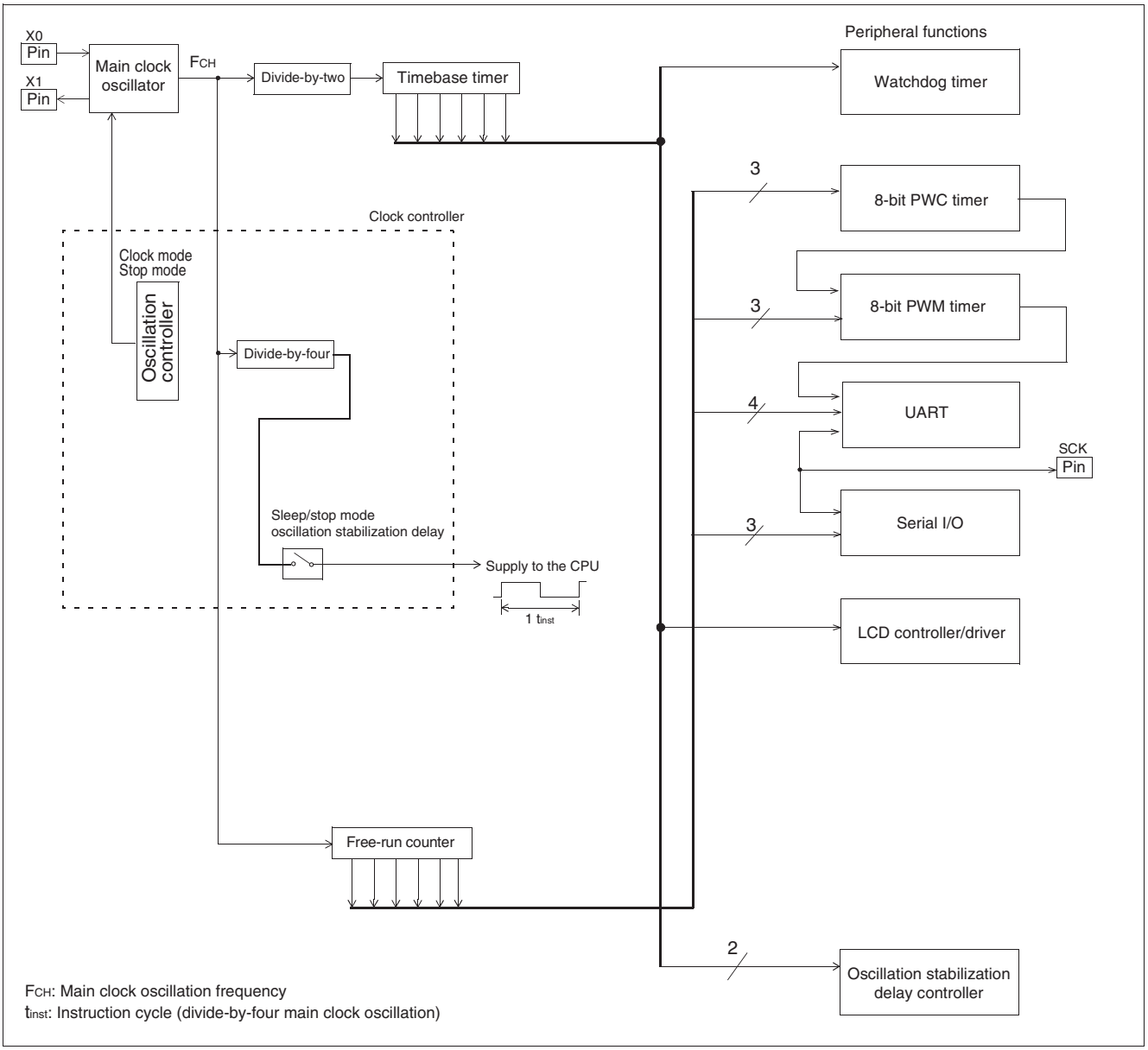
Divide-by-n output derived from the free-run counter clocked by the peripheral circuit clock is supplied to the peripheral functions.

Divide-by-n outputs from the timebase timer are also supplied to the peripheral functions.

These clocks, however, are not affected by the speed-shift function, etc. The timebase timer is clocked by the output of the main clock source oscillator after it is fed through a divide-by-2 circuit.

Figure 3.6-1 "Clock supply map" shows the clock supply map.

Figure 3.6-1 Clock supply map



3.6.1 Clock Generator

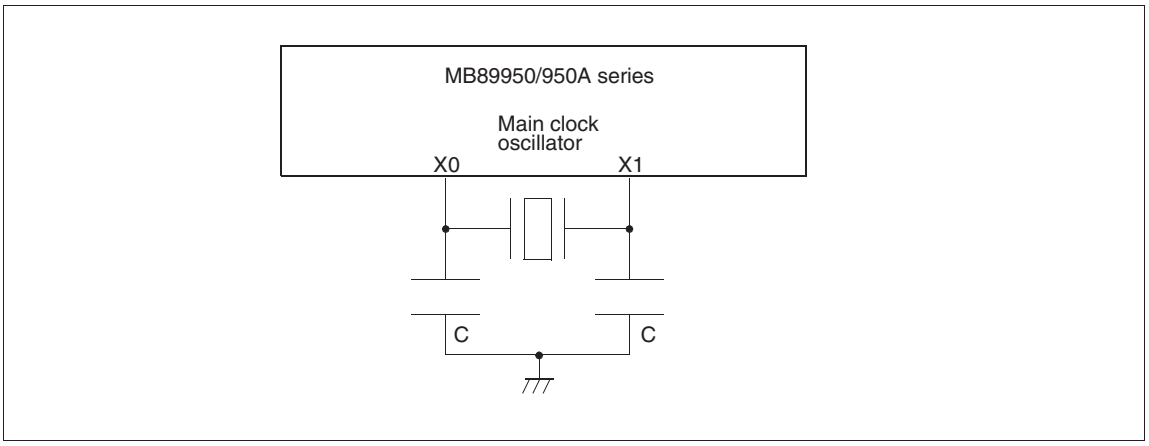
Enable and stop of the main clock oscillation are controlled by clock and stop mode respectively.

■ Clock generator

● Crystal or ceramic resonator

Connect as shown in Figure 3.6-2 "Connection example for a crystal or ceramic resonator".

Figure 3.6-2 Connection example for a crystal or ceramic resonator



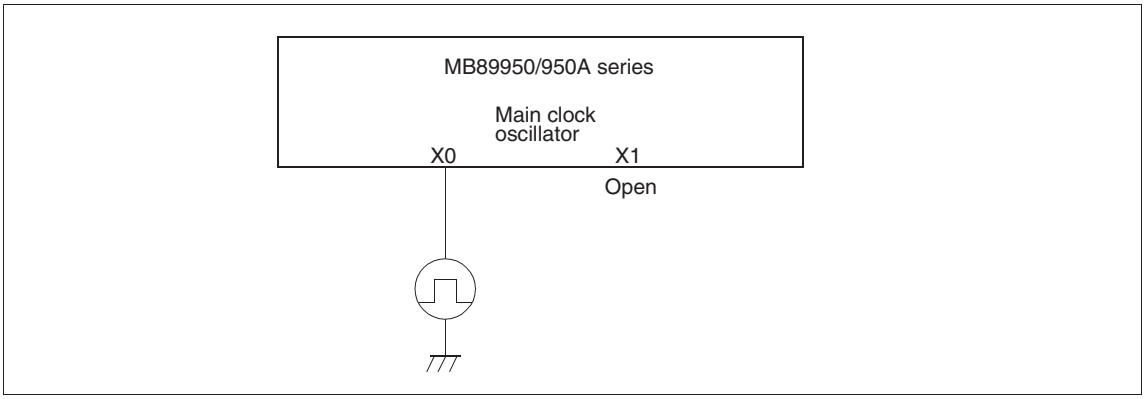
Reference:

A piezoelectric resonator (FAR series) that contains the external capacitors can also be used. See Data Sheet for details.

● External clock

Connect the external clock to the X0 pin and leave X1 pin open, as shown in Figure 3.6-3 "Connection example for external clock".

Figure 3.6-3 Connection example for external clock



3.6.2 Clock Controller

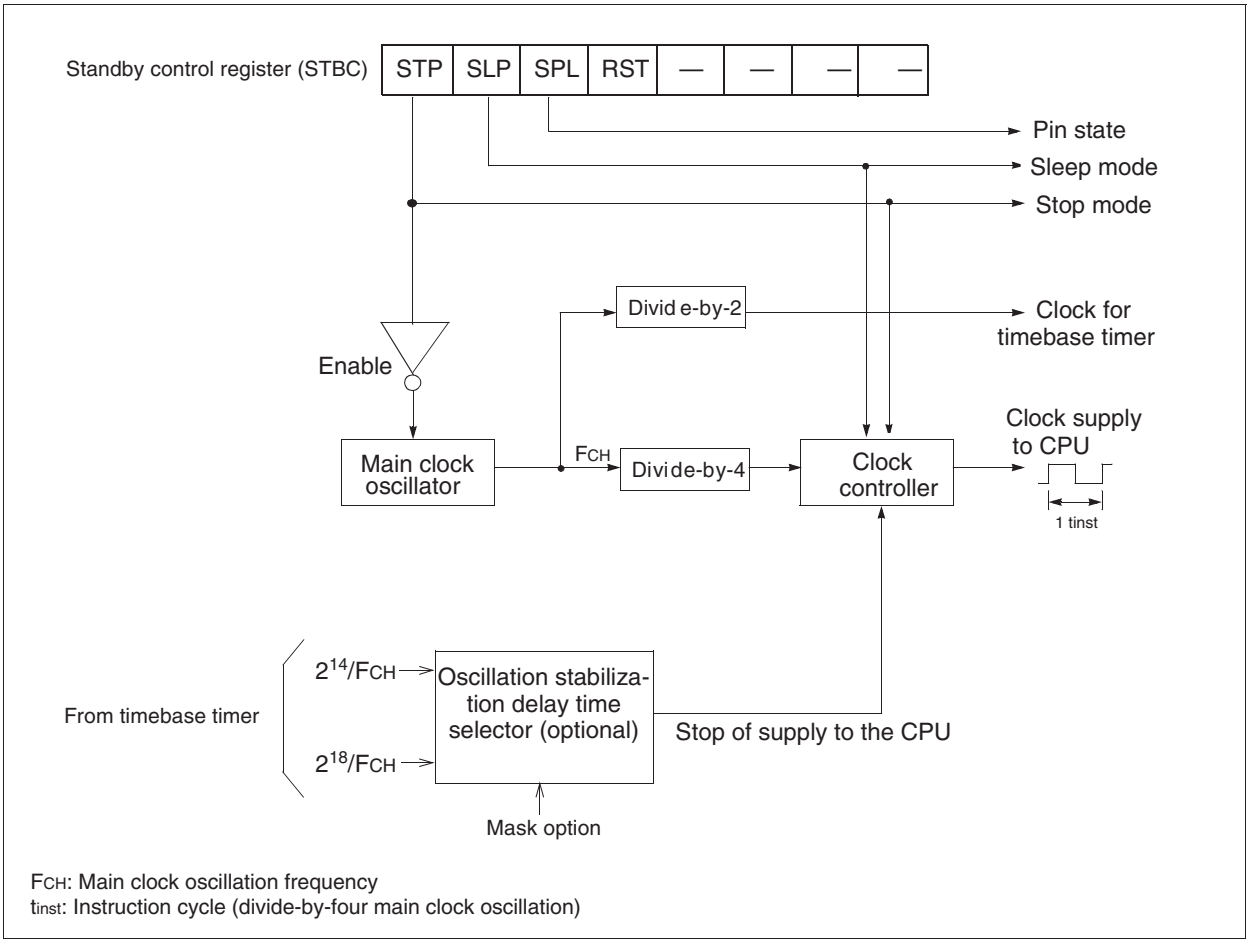
The clock controller contains the following four blocks:

- **Main clock oscillator**
- **Clock controller**
- **Oscillation stabilization delay time selector**
- **Standby control register (STBC)**

■ Block diagram of clock controller

Figure 3.6-4 "Block diagram of clock controller" shows the block diagram of the clock controller.

Figure 3.6-4 Block diagram of clock controller



● Main clock oscillator

The main clock oscillator is stopped in main stop mode.

- Clock controller

 This circuit controls the supply of operating clocks to the CPU and peripheral circuits, selecting the clock based on the active mode: normal (RUN), or standby (sleep/stop) mode.

 Supply of the clock to the CPU is stopped until the clock supply stop signal in the oscillation stabilization delay time selector is released.

- Oscillation stabilization delay time selector

 This selector selects a delay time between two main clock oscillation stabilization times timed by the timebase timer as the duration of CPU clock stop signal.

- STBC register

 This register controls from normal operation (RUN) to the standby mode, sets the pin states in the stop mode, and initiates software reset.

■ Instruction cycle (t_{inst})

Instruction cycle (minimum execution time) is 1/4 of the main clock.

3.6.3 Oscillation Stabilization Delay Time

When the system goes to run mode from a state in which the main clock is stopped (such as at power-on, and in stop mode and etc.), a delay time is required for oscillation to stabilize before starting any operation.

■ Oscillation stabilization delay time

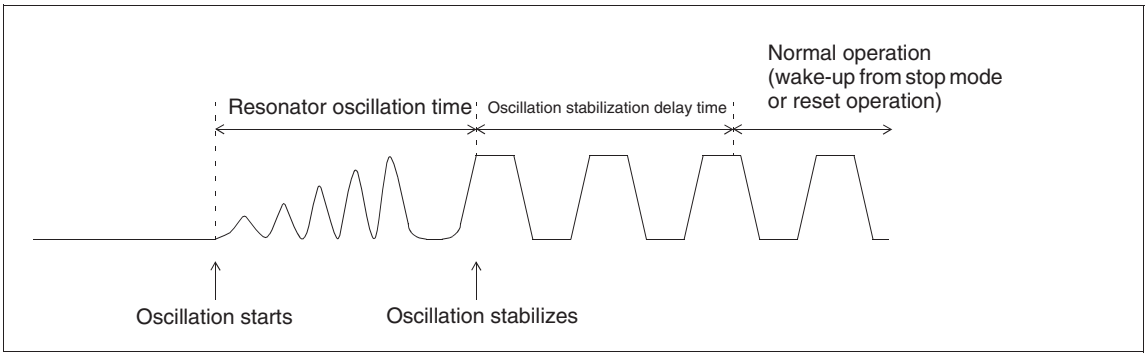
After starting, ceramic, crystal, and other resonators typically require the time between several milliseconds and several tens of milliseconds to stabilize at their fixed oscillation frequency.

Therefore, operation of the CPU and other functions is disabled when oscillation first starts and no clock signal is supplied to the CPU and peripheral functions until the oscillation stabilization delay time has passed and the oscillation has sufficiently stabilized.

The time required for oscillation to stabilize depends on the resonator type (crystal, ceramic, etc.) connected to the clock generator. Consequently, it is necessary to select an oscillation stabilization delay time that matches the type of oscillator being used.

Figure 3.6-5 "Operation of oscillator after starting oscillation" shows the operation of an oscillator after starting oscillation.

Figure 3.6-5 Operation of oscillator after starting oscillation



■ Main clock oscillation stabilization delay time

When first starting operation in main clock mode after a state in which the main clock oscillator is stopped, a delay time is required for oscillation to stabilize. This delay time starts when the timebase timer starts counting up from its cleared state, and ends when the count overflows at the specified bit.

● Oscillation stabilization delay time during operation

A time length must be selected for the oscillation stabilization delay time when an external interrupt takes the system from stop mode back to run mode. One of two possible delay times can be selected by mask option.

● Oscillation stabilization delay time at reset

The oscillation stabilization delay time at reset (the initial values of WT1 and WT0) is selected as an option setting.

Products with power-on reset require an oscillation stabilization delay time when exit from stop mode is triggered by resets in power-on reset, or external reset.

Table 3.6-1 "Main clock startup conditions vs. oscillation stabilization delay time" shows the relationships between the conditions in which main clock mode operation is started and oscillation stabilization delay time.

Table 3.6-1 Main clock startup conditions vs. oscillation stabilization delay time

Main clock mode startup conditions	At power-on	Exit from stop mode	
		External reset	External interrupt
Oscillation stabilization delay time selection	Option setting		
With power-on reset	○	○	○
No power-on reset	X	X	○

○: Oscillation stabilization delay time provided
X: Oscillation stabilization delay time not provided

3.7 Standby Mode (Low-power Consumption)

The standby mode consists of sleep mode and stop mode.
Main run mode is switched to sleep mode or stop mode by setting the standby control register (STBC).
Standby mode reduces the power consumption by stopping the operation of the CPU and peripheral functions.
This section describes the relationship between standby mode and clock mode, and the operation of various sections during standby.

■ Standby mode

Standby mode reduces the power consumption, however, by stopping the clock signal supply to the CPU via clock controller (sleep mode), or by stopping the source oscillator itself (stop mode).

● Sleep mode

Sleep mode stops the CPU and watchdog timer, but operate the peripheral functions.

● Stop mode

Stop mode stops the CPU and peripheral functions. The main clock oscillator is stopped. Everything is shut down except external interrupt service.

3.7.1 Operating States in Standby Mode

This section describes the operating states of the CPU and peripheral functions in standby mode.

■ Operating states during standby mode

Table 3.7-1 Operating states of the CPU and peripheral functions in standby mode

Function		Main clock mode			
		Run	Sleep	Stop (SPL = "0")	Stop (SPL = "1")
Main clock		Operating	Operating	Stop	Stop
CPU	Instructions	Operating	Stop	Stop	Stop
	ROM	Operating	Hold	Hold	Hold
	RAM				
Peripheral functions	I/O ports	Operating	Hold	Hold	Hi-Z (*1)
	Timebase timer	Operating	Operating	Stop	Stop
	8-bit PWM timer	Operating	Operating	Stop	Stop
	8-bit PWC timer	Operating	Operating	Stop	Stop
	Watchdog timer	Operating	Stop	Stop	Stop
	LCD controller/driver	Operating	Operating	Stop	Stop
	External Interrupts	Operating	Operating	Operating	Operating
	Serial I/O	Operating	Operating	Stop	Stop
	UART	Operating	Operating	Stop	Stop

*1: If pull-up is selected, it will be "H" level.

● Pin States in Standby Mode

Almost all I/O pins will either keep the state they were placed in, or go to the high-impedance state according to the pin state control bit of the standby control register (STBC: SPL) just prior to going to the stop mode. This is true regardless of the clock mode.

See Appendix E "MB89950/950A Series Pin States" for pin states in standby mode.

3.7.2 Sleep Mode

This section describes the operations of sleep mode.

■ Operation of sleep mode

● Entering sleep mode

Sleep mode stops the CPU operating clock. The CPU stops while maintaining all register contents, RAM contents, and pin states at their values immediately prior to entering sleep mode. However, peripheral functions except the watchdog timer continue to operate.

Writing "1" to the sleep bit in the standby control register (STBC: SLP) puts the CPU to sleep mode. If an interrupt request is generated when "1" is written to the SLP bit, the write to the bit is ignored, and the CPU continues the instruction execution without entering sleep mode. (The CPU does not go to sleep mode even after completion of the interrupt processing.)

● Wake-up from sleep mode

A reset or an interrupt from a peripheral function wakes up the CPU from sleep mode.

There is no oscillation stabilization delay period.

The reset operation also initializes the pin states.

If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function or an external interrupt circuit during sleep mode, the CPU wakes up from sleep mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1 and IL0) in the CPU.

The normal interrupt operation is performed after wake-up from sleep mode. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering sleep mode.

3.7.3 Stop Mode

This section describes the operations of stop mode.

■ Operation of stop mode

● Entering stop mode

Stop mode stops the oscillation source. Almost all functions stop while maintaining all register and RAM contents at their value immediately before entering stop mode.

Writing "1" to the stop bit in the standby control register (STBC: STP) puts the CPU to stop mode. At this time, external pin states are held if the pin state specification bit (STBC: SPL) is "0". If SPL is "1", external pins go to the high-impedance state. (Pins with the pull-up resistor (optional) go to the "H" level.)

If an interrupt request is generated when "1" is written to the STP bit, the write to the bit is ignored, and the CPU continues the instruction execution without entering stop mode. (The CPU does not assume stop mode even after completion of the interrupt processing.)

Prohibit interrupt request output from the timebase timer (TBTC: TBIE = "0") before entering stop mode in main clock mode as necessary.

● Wake-up from stop mode

A reset or an external interrupt wakes up the CPU from stop mode.

If reset occurs during stop mode on a product with power-on reset, the reset operation starts after the main clock oscillation stabilization delay time. Products without power-on reset do not require for the oscillation stabilization delay time after a reset in stop mode. The reset initializes pin states.

If an interrupt request with an interrupt level higher than "11_B" occurs from an external interrupt circuit during stop mode, the CPU wakes up from stop mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1, IL0) in the CPU. Only external interrupt requests can occur during stop mode because peripheral functions are stopped.

After wake-up from stop mode, the normal interrupt operation is performed after the oscillation stabilization delay time has passed. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering stop mode.

Some peripheral functions restart from mid-operation when the CPU wakes up from stop mode by an external interrupt. The first interval time from the interval timer function, for example, is indeterminate. Therefore, initialize all peripheral functions after wake-up from stop mode.

Note:

Only interrupt requests from external interrupt circuits can be used to wake up from stop mode by an interrupt.

3.7.4 Standby Control Register (STBC)

The standby control register (STBC) controls the CPU to enter to sleep mode, stop mode, sets the pin states in stop mode, and initiates software reset.

■ Standby control register (STBC)

Figure 3.7-1 Standby control register (STBC)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0008H	STP	SLP	SPL	RST	—	—	—	—	0001----B
	W	W	R/W	W					

RST	Software reset bit	
	Read	Write
0	—	Generates a reset signal for four instruction cycles.
1	Reading always returns “1”.	No effect on operation.

SPL	Pin state specification bit
0	External pins hold their states prior to entering stop mode.
1	External pins go to high-impedance state on entering stop mode.

SLP	Sleep bit	
	Read	Write
0	Reading always returns “0”.	No effect on operation.
1	—	Goes to sleep mode.

STP	Stop bit	
	Read	Write
0	Reading always returns “0”.	No effect on operation.
1	—	Goes to stop mode.

R/W : Readable and writable
W : Write-only
— : Unused
X : Indeterminate
: Initial value

Table 3.7-2 Standby control register (STBC) bits

Bit		Function
Bit 7	STP: Stop bit	• Sets the CPU entering stop mode. • Writing "1" to this bit sets the CPU entering stop mode. • Writing "0" to this bit has no effect on operation. • Reading this bit always returns "0".
Bit 6	SLP: Sleep bit	• Sets the CPU entering sleep mode. • Writing "1" to this bit sets the CPU entering sleep mode. • Writing "0" to this bit has no effect on operation. • Reading this bit always returns "0".
Bit 5	SPL: Pin state specification bit	• Specifies the states of the external pins during stop mode. • Writing "0" to this bit specifies that external pins hold their states (levels) when entering stop mode. • Writing "1" to this bit specifies that external pins go to high-impedance state when entering stop mode (pin with a pull-up resistor (optional) go to "H" level). • Initialized to "0" by a reset.
Bit 4	RST: Software reset bit	• Specifies a software reset. • Writing "0" to this bit generates an internal reset source for four instruction cycles. • Writing "1" to this bit has no effect on operation. • Reading this bit always returns "1".
Bit 3 Bit 2 Bit 1 Bit 0	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on operation.

3.7.5 State Transition Diagram

This section shows two state transition diagrams: one diagram for "with power-on reset" option products and the other for "without power-on reset" products.

■ State transition diagrams

Figure 3.7-2 State transition diagram (products with power-on reset)

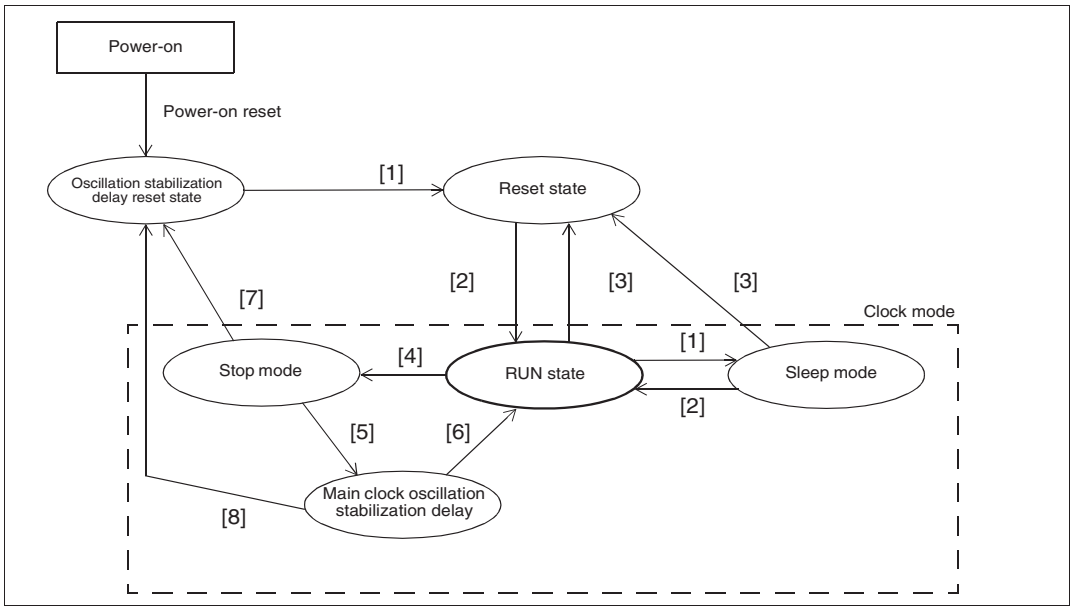
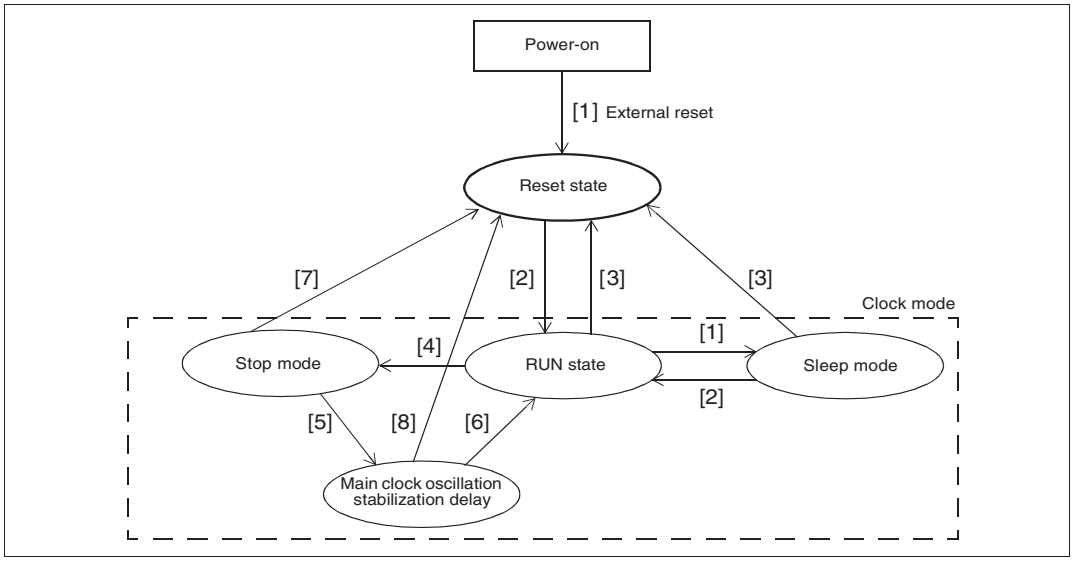


Figure 3.7-3 State transition diagram (products without power-on reset)



● Go to normal state (RUN) and reset

Table 3.7-3 Go to main clock mode run state and reset

State transition	Conditions/events required for transition	
	Products with power-on reset (Figure 3.7-2)	Products without power-on reset (Figure 3.7-3)
Go to normal state (RUN) after power-on	[1] Main clock oscillation stabilization delay time completes (timebase timer output). [2] Wake-up from Reset input.	[1] External reset input must be held asserted until main clock oscillation has had time to stabilize. [2] Wake-up from reset input de-asserted.
Reset in RUN state	[3] Have external, software, or watchdog reset.	[3] Have external, software, or watchdog reset.

● Go to/wake-up from standby mode

Table 3.7-4 Go to/wake-up from standby mode

State transition	Conditions/events required for transition	
	Products with power-on reset (Figure 3.7-2)	Products without power-on reset (Figure 3.7-3)
Go to sleep mode	[1] STBC: SLP = "1"	[1] STBC: SLP = "1"
Wake-up from sleep mode	[2] Interrupt [3] External reset	[2] Interrupt [3] External reset
Go to stop mode	[4] STBC: STP = "1"	[4] STBC: STP = "1"
Wake-up from stop mode	[5] External interrupt [6] Main clock oscillation stabilization delay time completes (timebase timer output). [7] External reset [8] External reset (during oscillation stabilization delay time)	[5] External interrupt [6] Main clock oscillation stabilization delay time completes (timebase timer output). [7] External reset [8] External reset (during oscillation stabilization delay time)

STBC: Standby control register

3.7.6 Notes on Using Standby Mode

The CPU does not go to standby mode if an interrupt request occurs from a peripheral function when a standby mode bit is set in the standby control register (STBC). Also, if an interrupt is used to wake up from a standby mode to the normal operating state, the operation after wake-up differs depending on whether or not the interrupt request is accepted.

■ Go to standby mode and interrupts

If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function to the CPU, writing "1" to the stop bit (STP), sleep bit (SLP) in the standby control register (STBC) is ignored. Therefore, the CPU does not go to standby mode (The CPU also does not go to the standby mode after completing interrupt processing). This does not depend on whether or not the CPU accepts the interrupt.

Even if the CPU is currently performing interrupt processing, after clearing the interrupt request flag bit the device can go to the standby mode if no other interrupt request is present.

■ Wake-up from standby mode by interrupt

If an interrupt request with an interrupt level higher than "11_B" occurs from a peripheral function or others during sleep or stop mode, the CPU wakes up from standby mode. This does not depend on whether or not the CPU accepts the interrupt.

After wake-up from standby mode, the CPU performs the normal interrupt operations. If the level set in the interrupt level setting register (ILR1 to ILR3) corresponding to the interrupt request is higher than the interrupt level bits in the condition code register (CCR: IL1, IL0), and if the interrupt enable flag is enabled (CCR: I = "1"), the CPU branches to the interrupt processing routine. If the interrupt is not accepted, operation restarts from the instruction following the instruction that activated the standby mode.

To prevent control from branching to an interrupt processing routine after wake-up, take measures such as disabling interrupts before setting standby mode bit.

■ Notes on setting standby mode

When setting the standby control register (STBC) to go to standby mode, make the settings in accordance with Table 3.7-5 "Standby control register (STBC) low-power consumption mode settings". Although the order of precedence as to which mode will be activated if more than one bit is set to "1" is stop mode and sleep mode, it is best to set "1" for just one bit.

Table 3.7-5 Standby control register (STBC) low-power consumption mode settings

STBC register		Mode
STP (Bit 7)	SLP (Bit 6)	
0	0	Normal
0	1	Sleep
1	0	Stop

■ Oscillation stabilization delay time

As the oscillator that provides the oscillation source is stopped during stop mode, a delay time is required for oscillation to stabilize after the oscillator restarts operation.

In main clock mode, the main clock oscillation stabilization delay time is selected from one of two possible delay times defined by the timebase timer.

In main clock mode, if the interval time set for the timebase timer is less than the oscillation stabilization delay time, the timebase timer generates an interval timer interrupt request before the end of the oscillation stabilization delay time. To prevent this, disable the interrupt request output for the timebase timer (TBTC: TBIE = "0") before going to stop mode in main clock mode as necessary.

3.8 Memory Access Mode

In the MB89950/950A series, the only memory access mode is the single-chip mode.

■ Single-chip mode

In single-chip mode, the device uses internal RAM and ROM only. Therefore, the CPU can access no areas other than the internal I/O area, RAM area, and ROM area (internal access).

■ Mode pin (MODA)

Always set the mode pin, MODA, to V_{SS}.

At reset, reads the mode data and reset vector from internal ROM.

Do not change the mode pin settings, even after completion of the reset (i.e. during normal operation).

Table 3.8-1 "Mode pin setting" lists the mode pin settings.

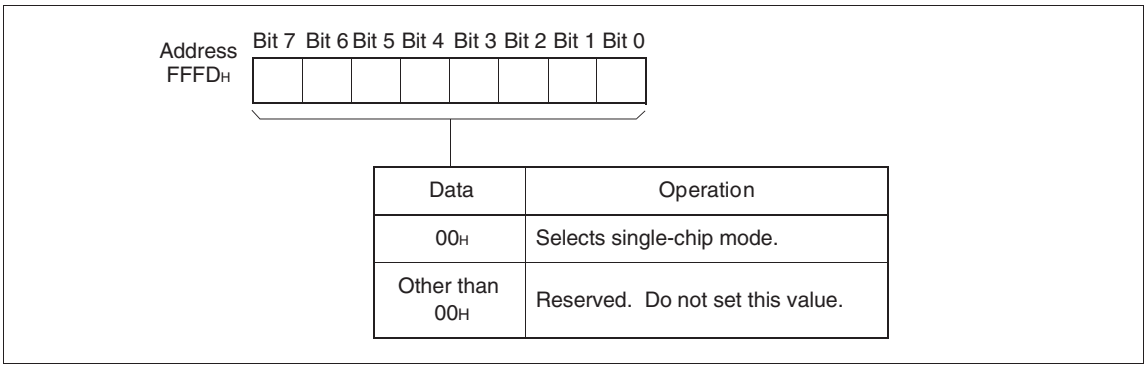
Table 3.8-1 Mode pin setting

MODA pin state	Description
V_{SS}	Reads the mode data and reset vector from internal ROM.
V_{CC}	Prohibited settings

■ Mode data

Always set the mode data in internal ROM to "00_H" to select single-chip mode.

Figure 3.8-1 Mode data structure



■ Memory access mode selection operation

Only the single-chip mode can be selected.

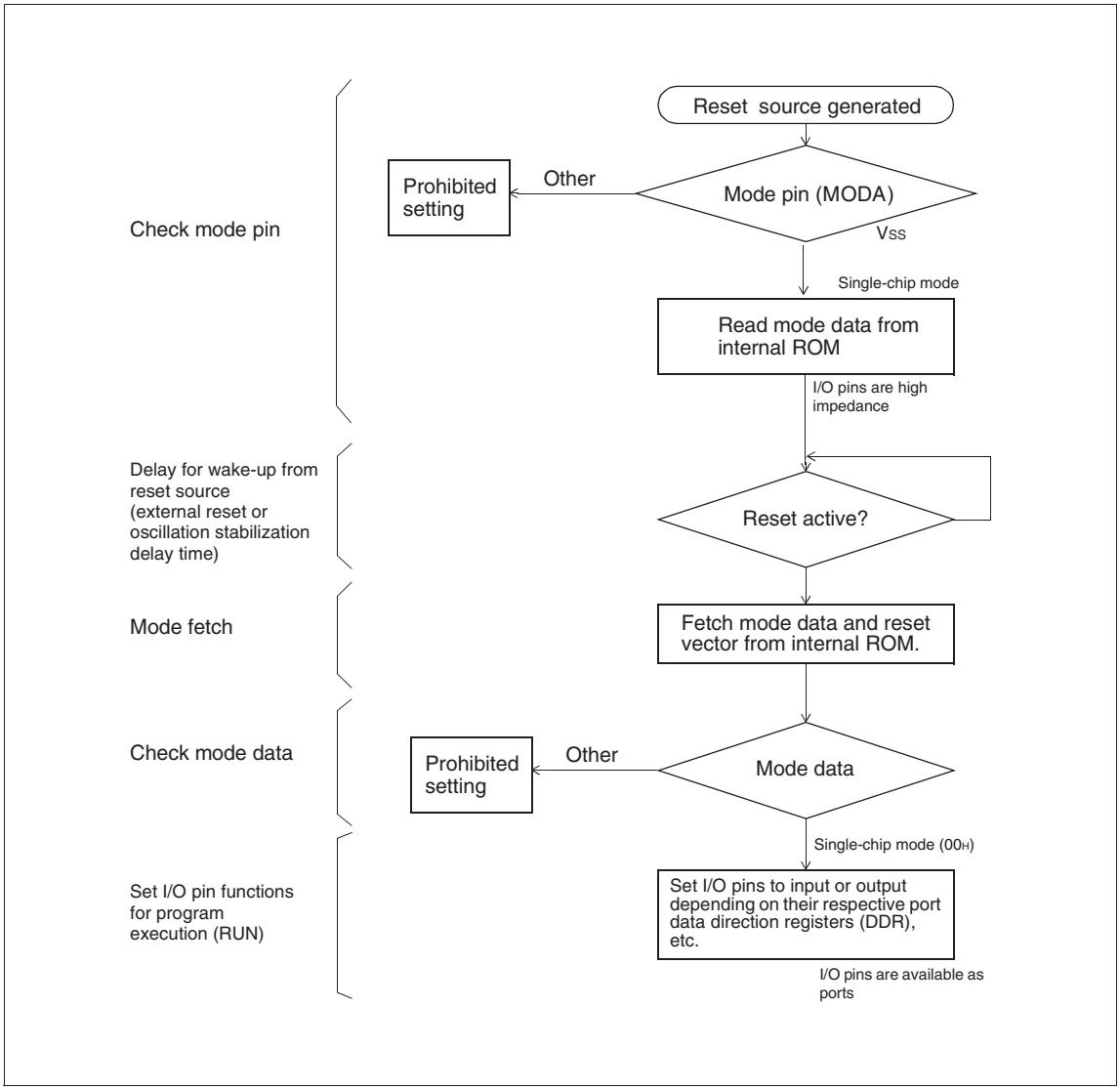
Table 3.8-2 "Mode pin and mode data" lists the mode pin and mode data options.

Table 3.8-2 Mode pins and mode data

Memory access mode	Mode pin (MODA)	Mode data
Single-chip mode	V_{SS}	00_H
Other modes	Prohibited settings	Prohibited settings

Figure 3.8-2 "Memory access selection operation" shows the operation for memory access mode selection.

Figure 3.8-2 Memory access selection operation



CHAPTER 4
I/O PORTS

This chapter describes the functions and operation of the I/O ports.

4.1 "Overview of I/O Ports"

4.2 "Port 0"

4.3 "Port 1"

4.4 "Port 2"

4.5 "Port 3"

4.6 "Port 4"

4.7 "Program Example for I/O Ports"

4.1 Overview of I/O Ports

The I/O ports consist of five ports (33 pins) including N-ch open-drain and CMOS general-purpose I/O ports (parallel I/O ports).
The ports also serve as peripherals (I/O pins of peripheral functions).

■ I/O port functions

The functions of the I/O ports are to output data from the CPU via the I/O pins and to fetch signals input to the I/O pins into the CPU. Input and output are performed via the port data registers (PDR). Also, for certain ports the direction of each I/O pin can be individually set to either input or output for each bit by the port data direction register (DDR).

The following lists the functions of each port and the peripheral with which the ports also serve as.

- Port 0: General-purpose N-ch open-drain I/O port. Also serves as LCD segment driver pins.
- Port 1: General-purpose N-ch open-drain I/O port. Also serves as LCD segment driver pins.
- Port 2: General-purpose N-ch open-drain I/O port. Also serves as LCD segment driver pins.
- Port 3: General-purpose N-ch open-drain I/O port. Also serves as LCD bias pins.
- Port 4: General-purpose CMOS I/O port. Also serves as other peripheral I/O pins.

Table 4.1-1 "Port function" lists the functions of each port and Table 4.1-2 "Port registers" lists the registers for each port.

Table 4.1-1 Port function

Port	Pin name	Input type	Output type	Function	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Port 0	P00/SEG20 to P07/SEG27	CMOS	N-ch open-drain	General-purpose I/O port	P07	P06	P05	P04	P03	P02	P01	P00
				Segment driver output	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20
Port 1	P10/SEG28 to P17/SEG35			General-purpose I/O port	P17	P16	P15	P14	P13	P12	P11	P10
				Segment driver output	SEG35	SEG34	SEG33	SEG32	SEG31	SEG30	SEG29	SEG28
Port 2	P20/SEG36 to P25/SEG41			General-purpose I/O port	--	--	P25	P24	P23	P22	P21	P20
				Segment driver output	--	--	SEG41	SEG40	SEG39	SEG38	SEG37	SEG36
Port 3	P30 to P33/V2			General-purpose I/O port	--	--	--	--	P33	P32	P31	P30
				LCD bias	--	--	--	--	V2	V1	--	--
Port 4	P40 to P46/INT0	CMOS (resource: hysteresis)	CMOS (push-pull option)	General-purpose I/O port	--	P46	P45	P44	P43	P42	P41	P40
				Peripherals	--	INT0	SCK	SO	SI	PWC/ INT1	PWM	--

Table 4.1-2 Port registers

Register	Read/Write	Address	Initial value
Port 0 data register (PDR0)	R/W	0000_H	11111111_B
Port 1 data register (PDR1)	R/W	0002_H	11111111_B
Port 2 data register (PDR2)	R/W	0004_H	$--111111_B$
Port 3 data register (PDR3)	R/W	$000C_H$	$----1111_B$
Port 4 data register (PDR4)	R/W	$000E_H$	$-XXXXXXX_B$
Port 4 data direction register (DDR4)	W	$000F_H$	-0000000_B

R/W: Readable and writable
W: Write-only
X: Indeterminate
-: Unused

4.2 Port 0

Port 0 is N-ch open-drain I/O port that also serves as LCD segment driver outputs. Port 0 pins can be switched between LCD segment driver output and port operation by mask option. This section principally describes the port functions when operating as N-ch open-drain I/O port.
The section describes the port structure and pins, the pin block diagram, and the port register for port 0.

■ Structure of port 0

Port 0 consists of the following two components:

- N-ch open-drain I/O pins/LCD segment driver output pins (P00/SEG20 to P07/SEG27)
- Port 0 data register (PDR0)

■ Port 0 pins

Port 0 consists of eight N-ch open-drain I/O. When pins are used by the peripheral, they cannot be used as N-ch open-drain I/O.

Table 4.2-1 "Port 0 pins" lists the port 0 pins.

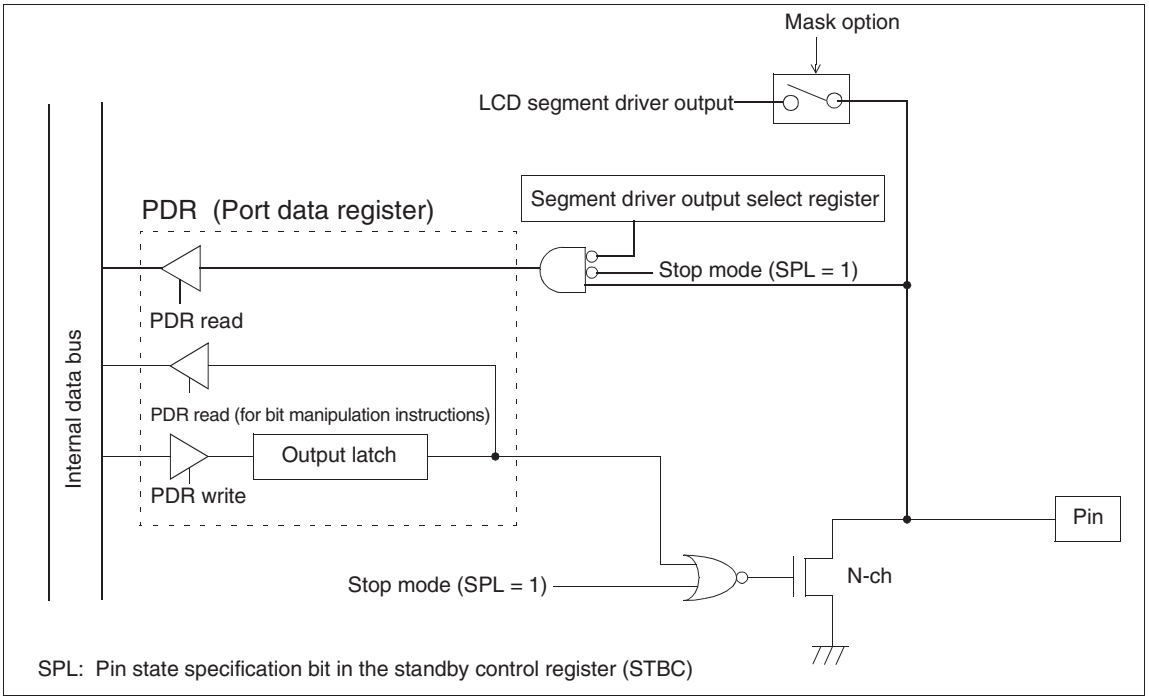
Table 4.2-1 Port 0 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 0	P00/SEG20	P00 N-ch open-drain I/O	SEG20 LCD segment driver output	CMOS	Segment / N-ch open-drain	D
	P01/SEG21	P01 N-ch open-drain I/O	SEG21 LCD segment driver output			
	P02/SEG22	P02 N-ch open-drain I/O	SEG22 LCD segment driver output			
	P03/SEG23	P03 N-ch open-drain I/O	SEG23 LCD segment driver output			
	P04/SEG24	P04 N-ch open-drain I/O	SEG24 LCD segment driver output			
	P05/SEG25	P05 N-ch open-drain I/O	SEG25 LCD segment driver output			
	P06/SEG26	P06 N-ch open-drain I/O	SEG26 LCD segment driver output			
	P07/SEG27	P07 N-ch open-drain I/O	SEG27 LCD segment driver output			

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of port 0 pins

Figure 4.2-1 Block diagram of port 0 pins



■ Port 0 register

The port 0 register consists of PDR0. Each bit in the register has a one-to-one relationship with a port 0 pin.

Table 4.2-2 "Correspondence between pin and register for port 0" shows the correspondence between the pins and register for port 0.

Table 4.2-2 Correspondence between pin and register for Port 0

Port	Correspondence between register bit and pin								
Port 0	PDR0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	P07	P06	P05	P04	P03	P02	P01	P00

4.2.1 Port 0 Data Register (PDR0)

This section describes the port 0 data register.

■ Port 0 data register functions

● Port 0 data register (PDR0)

The PDR0 register holds the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Reference:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Settings as an LCD segment driver output

To use pins as LCD segment driver outputs, segment driver output must be selected by the mask option. Furthermore, the segment driver output select register must be set to the same as the mask option, so that the CMOS input port can be protected.

Table 4.2-3 "Port 0 data register function" a lists the functions of the port 0 data register.

Table 4.2-3 Port 0 data register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 0 data register (PDR0)	0	Pin state is the "L" level.	Outputs an "L" level to the pin. (Sets "0" to the output latch and turn the output transistor "ON".)	R/W	0000_H	11111111_B
	1	Pin state is the "H" level.	Sets the pin to the high-impedance state. (Sets "1" to the output latch and turn the output transistor "OFF".)			

R/W: Readable and writable

4.2.2 Operation of Port 0

This section describes the operations of the port 0.

■ Operation of port 0

● Operation as an output port

- When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and high impedance (Hi-Z) is output to the pin.
- Writing data to the PDR0 register stores the data in the output latch and it will be output to the pin.
- Reading the PDR0 register returns the output latch value.

● Operation as an input port

- Writing "0" to the PDR0 register set the port as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Reading the PDR0 register returns the pin value.

● Operation as an LCD segment driver output

- When the LCD output mask option is selected, set the PDR0 register bits corresponding to the LCD segment driver output pins to "1" to turn the output transistor "OFF".
- You cannot read the LCD output data by reading PDR0.

● Operation at reset

- Resetting the CPU initializes the PDR0 register values to "1". This turns "OFF" the output transistor for all pins and all pins are in high-impedance (Hi-Z) state.

● Operation in stop mode

- The output transistors are forcibly turned "OFF" regardless of the PRD0 register value and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device goes to stop mode. Moreover, to avoid leakage (from floating input pin), input must be driven by either "1" or "0" when SPL = "1".

Table 4.2-4 "Port 0 pin state" lists the port 0 pin states

Table 4.2-4 Port 0 pin state

Pin name	Normal operation sleep mode stop mode (SPL = "0")	Stop mode (SPL = "1")	Reset
P00/SEG20 to P07/SEG27	General-purpose I/O ports/segment driver output	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

4.3 Port 1

Port 1 is N-ch open-drain I/O port that also serves as LCD segment driver outputs. Port 1 pins can be switched between LCD segment driver output and port operation by mask option. This section principally describes the port functions when operating as N-ch open-drain I/O port.
The section describes the port structure and pins, the pin block diagram, and the port register for port 1.

■ Structure of port 1

Port 1 consists of the following two components:

- N-ch open-drain I/O pins/LCD segment driver output pins (P10/SEG28 to P17/SEG35)
- Port 1 data register (PDR1)

■ Port 1 pins

Port 1 consists of eight N-ch open-drain I/O. When pins are used by the peripheral, they cannot be used as N-ch open-drain I/O.

Table 4.3-1 "Port 1 pins" lists the port 1 pins.

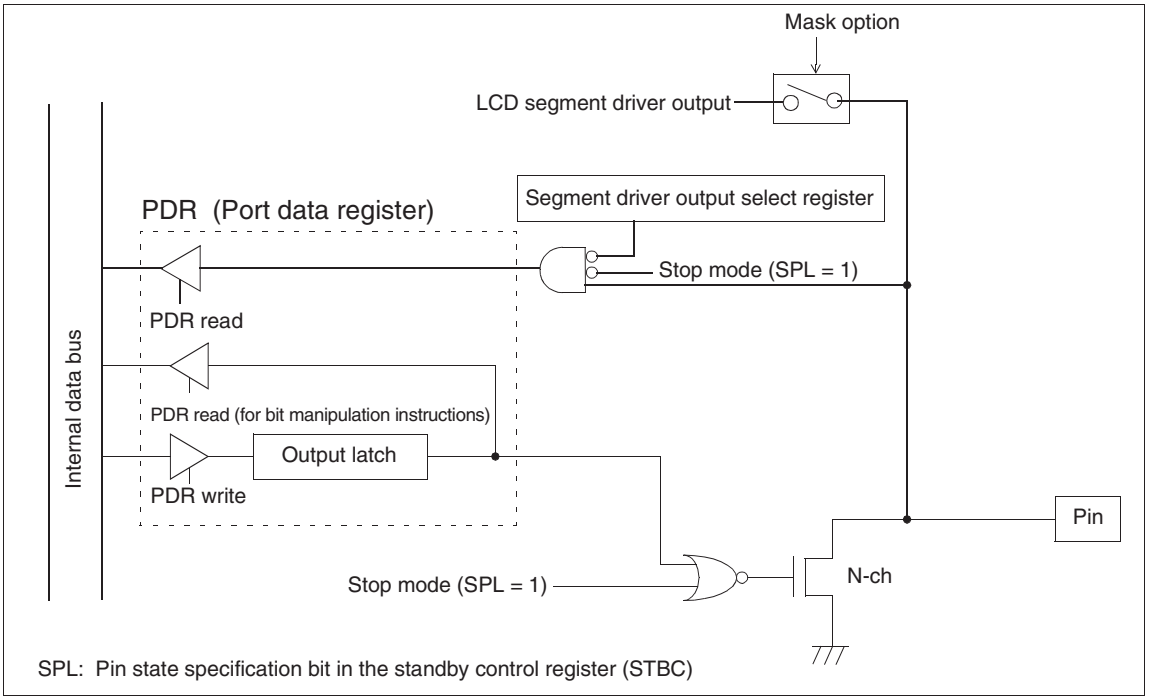
Table 4.3-1 Port 1 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 1	P10/SEG28	P10 N-ch open-drain I/O	SEG28 LCD segment driver output	CMOS	Segment / N-ch open-drain	D
	P11/SEG29	P11 N-ch open-drain I/O	SEG29 LCD segment driver output			
	P12/SEG30	P12 N-ch open-drain I/O	SEG30 LCD segment driver output			
	P13/SEG31	P13 N-ch open-drain I/O	SEG31 LCD segment driver output			
	P14/SEG32	P14 N-ch open-drain I/O	SEG32 LCD segment driver output			
	P15/SEG33	P15 N-ch open-drain I/O	SEG33 LCD segment driver output			
	P16/SEG34	P16 N-ch open-drain I/O	SEG34 LCD segment driver output			
	P17/SEG35	P17 N-ch open-drain I/O	SEG35 LCD segment driver output			

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of port 1 pins

Figure 4.3-1 Block diagram of port 1 pins



■ Port 1 register

The port 1 register consists of PDR1. Each bit in the register has a one-to-one relationship with a port 1 pin.

Table 4.3-2 "Correspondence between pin and register for port 1" shows the correspondence between the pins and register for port 1.

Table 4.3-2 Correspondence between pin and register for port 1

Port	Correspondence between register bit and pin								
Port 1	PDR1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	P17	P16	P15	P14	P13	P12	P11	P10

4.3.1 Port 1 Data Register (PDR1)

This section describes the port 1 data register.

■ Port 1 data register functions

● Port 1 data register (PDR1)

The PDR1 register holds the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read the output latch state.

Reference:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Settings as an LCD segment driver output

To use pins as LCD segment driver outputs, segment driver output must be selected by the mask option. Furthermore, the segment driver output select register must be set to the same as the mask option, so that the CMOS input port can be protected.

Table 4.3-3 "Port 1 data register function" lists the functions of the port 1 data register.

Table 4.3-3 Port 1 data register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 1 data register (PDR1)	0	Pin state is the "L" level.	Outputs an "L" level to the pin. (Sets "0" to the output latch and turn the output transistor "ON".)	R/W	0002_H	11111111_B
	1	Pin state is the "H" level.	Sets the pin to the high-impedance state. (Sets "1" to the output latch and turn the output transistor "OFF".)			

R/W: Readable and writable

4.3.2 Operation of Port 1

This section describes the operations of the port 1.

■ Operation of port 1

● Operation as an output port

- When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and high impedance (Hi-Z) is output from the pin.
- Writing data to the PDR1 register stores the data in the output latch and it will be output to the pin.
- Reading the PDR1 register returns the output latch value.

● Operation as an input port

- Writing "0" to the PDR1 register set the port as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Reading the PDR1 register returns the pin value.

● Operation as an LCD segment driver output

- When the LCD output mask option is selected, set the PDR1 register bits corresponding to the LCD segment driver output pins to "1" to turn the output transistor "OFF".
- You cannot read the LCD output data by reading PDR1.

● Operation at reset

- Resetting the CPU initializes the PDR1 register values to "1". This turns "OFF" the output transistor for all pins and all pins are in high impedance (Hi-Z) state.

● Operation in stop mode

- The output transistors are forcibly turned "OFF" regardless of the PRD0 register value and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device goes to stop mode. Moreover, to avoid leakage (from floating input pin), input must be driven by either "1" or "0" when SPL = "1".

Table 4.3-4 "Port 1 pin state" lists the port 1 pin states.

Table 4.3-4 Port 1 pin state

Pin name	Normal operation sleep mode stop mode (SPL = "0")	Stop mode (SPL = "1")	Reset
P10/SEG28 to P17/SEG35	General-purpose I/O ports/segment driver output	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

4.4 Port 2

Port 2 is N-ch open-drain I/O port that also serves as LCD segment driver outputs. Port 2 pins can be switched between LCD segment driver output and port operation by mask option. This section principally describes the port functions when operating as N-ch open-drain I/O port.
The section describes the port structure and pins, the pin block diagram, and the port register for port 2.

■ Structure of port 2

Port 2 consists of the following two components:

- N-ch open-drain I/O pins/LCD segment driver output pins (P20/SEG36 to P25/SEG41)
- Port 2 data register (PDR2)

■ Port 2 pins

Port 2 consists of six N-ch open-drain I/O. When pins are used by the peripheral, they cannot be used as N-ch open-drain I/O.

Table 4.4-1 "Port 2 pins" lists the port 2 pins.

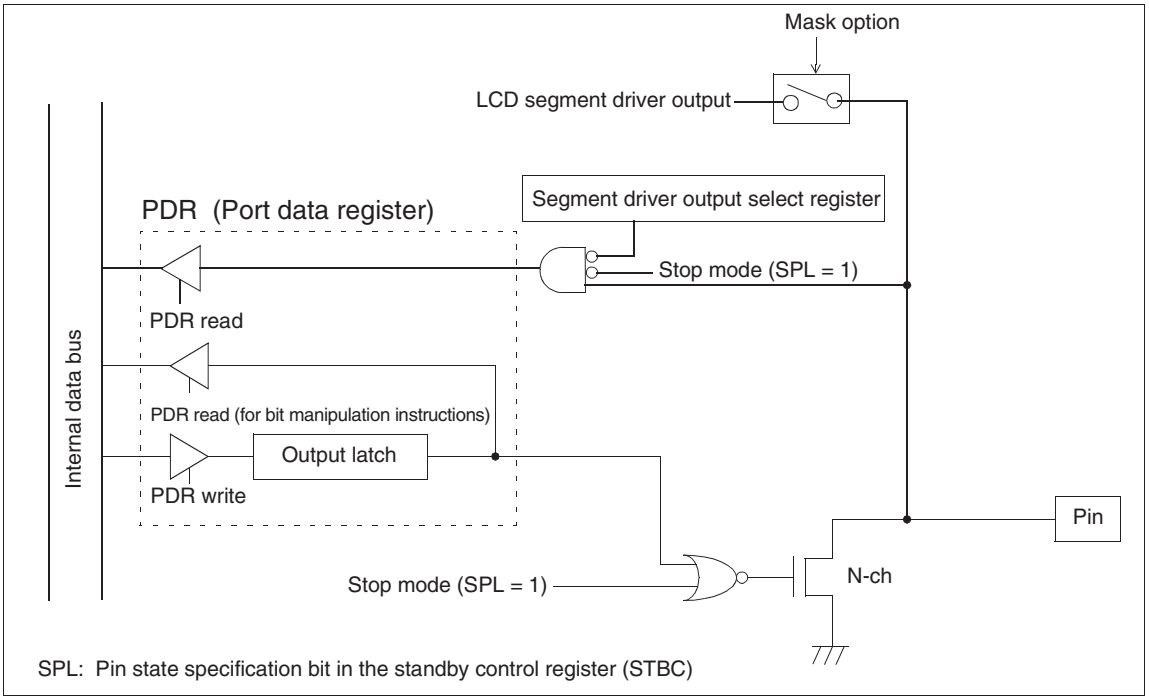
Table 4.4-1 Port 2 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 2	P20/SEG36	P20 N-ch open-drain I/O	SEG36 LCD segment driver output	CMOS	Segment / N-ch open-drain	D
	P21/SEG37	P21 N-ch open-drain I/O	SEG37 LCD segment driver output			
	P22/SEG38	P22 N-ch open-drain I/O	SEG38 LCD segment driver output			
	P23/SEG39	P23 N-ch open-drain I/O	SEG39 LCD segment driver output			
	P24/SEG40	P24 N-ch open-drain I/O	SEG40 LCD segment driver output			
	P25/SEG41	P25 N-ch open-drain I/O	SEG41 LCD segment driver output			

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of port 2 pins

Figure 4.4-1 Block diagram of port 2 pins



■ Port 2 register

The port 2 register consists of PDR2. Each bit in the register has a one-to-one relationship with a port 2 pin.

Table 4.4-2 "Correspondence between pin and register for port 2" shows the correspondence between the pins and register for port 2.

Table 4.4-2 Correspondence between pin and register for port 2

Port	Correspondence between register bit and pin								
Port 2	PDR2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	--	--	P25	P24	P23	P22	P21	P20

4.4.1 Port 2 Data Register (PDR2)

This section describes the port 2 data register.

■ Port 2 data register functions

● Port 2 data register (PDR2)

The PDR2 register holds the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Reference:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Settings as an LCD segment driver output

To use pins as LCD segment driver outputs, segment driver output must be selected by the mask option. Furthermore, the segment driver output select register must be set to the same as the mask option, so that the CMOS input port can be protected.

Table 4.4-3 "Port 2 data register function" lists the functions of the port 2 data register.

Table 4.4-3 Port 2 data register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 2 data register (PDR2)	0	Pin state is the "L" level.	Outputs an "L" level to the pin. (Sets "0" to the output latch and turn the output transistor "ON".)	R/W	0004_H	$--111111_B$
	1	Pin state is the "H" level.	Sets the pin to the high-impedance state. (Sets "1" to the output latch and turn the output transistor "OFF".)			

R/W: Readable and writable
-: Unused bit

4.4.2 Operation of Port 2

This section describes the operations of the port 2.

■ Operation of port 2

● Operation as an output port

- Writing data to the PDR2 register stores the data in the output latch. When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and high impedance (Hi-Z) is output from the pin.
- Reading the PDR2 register returns the output latch value.

● Operation as an input port

- Writing "0" to the PDR2 register set the port as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Reading the PDR2 register returns the pin value.

● Operation as an LCD segment driver output

- When the LCD output mask option is selected, set the PDR2 register bits corresponding to the LCD segment driver output pins to "1" to turn the output transistor "OFF".
- You cannot read the LCD output data by reading PDR2.

● Operation at reset

- Resetting the CPU initializes the PDR2 register values to "1". This turns "OFF" the output transistor for all pins and all pins are in high-impedance (Hi-Z) state.

● Operation in stop mode

- The output transistors are forcibly turned "OFF" and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device goes to stop mode.

Table 4.4-4 "Port 2 pin state" lists the port 2 pin states.

Table 4.4-4 Port 2 pin state

Pin name	Normal operation sleep mode stop mode (SPL = "0")	Stop mode (SPL = "1")	Reset
P20/SEG36 to P25/SEG41	General-purpose I/O ports/segment driver output	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

4.5 Port 3

Port 3 is N-ch open-drain I/O port. Two of them also serve as LCD bias input. Port 3 pins can be switched between LCD bias input and port operation. This section principally describes the port functions when operating as N-ch open-drain I/O port.
The section describes the port structure and pins, the pin block diagram, and the port register for port 3.

■ Structure of port 3

Port 3 consists of the following two components:

- N-ch open-drain I/O pins/LCD bias input pins (P30 to P33/V2)
- Port 3 data register (PDR3)

■ Port 3 pins

Port 3 consists of four N-ch open-drain I/O. When pins are used by the peripheral, they cannot be used as N-ch open-drain I/O.

Table 4.5-1 "Port 3 pins" lists the port 3 pins.

Table 4.5-1 Port 3 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 3	P30	P30 N-ch open-drain I/O	--	CMOS	N-ch open-drain	F
	P31	P31 N-ch open-drain I/O	--			
	P32/V1	P32 N-ch open-drain I/O	V1 LCD bias input	LCD bias/CMOS	N-ch open-drain	H
	P33/V2	P33 N-ch open-drain I/O	V2 LCD bias input			

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of port 3 pins

Figure 4.5-1 Block diagram of port 3 pins (P30 and P31)

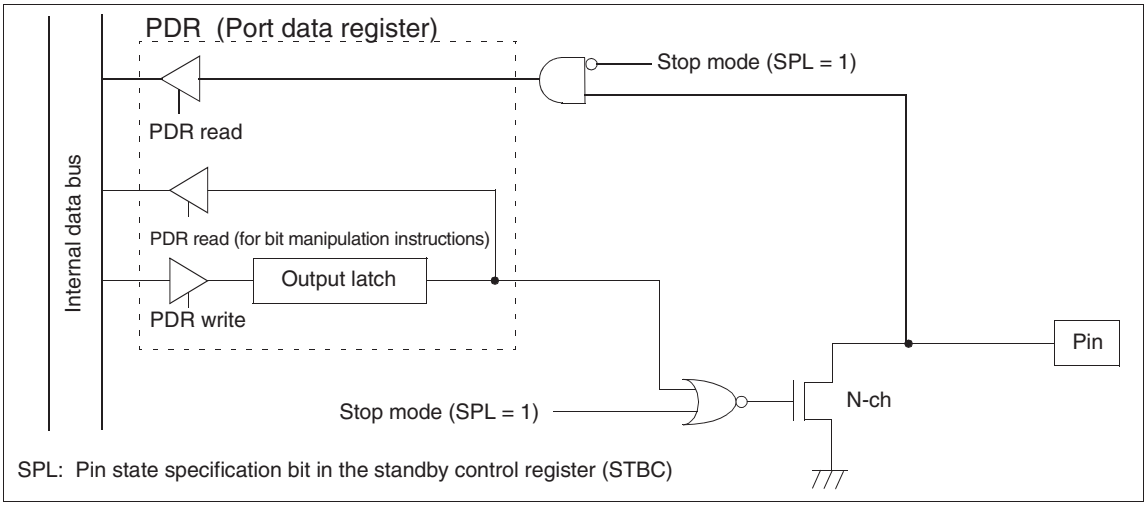
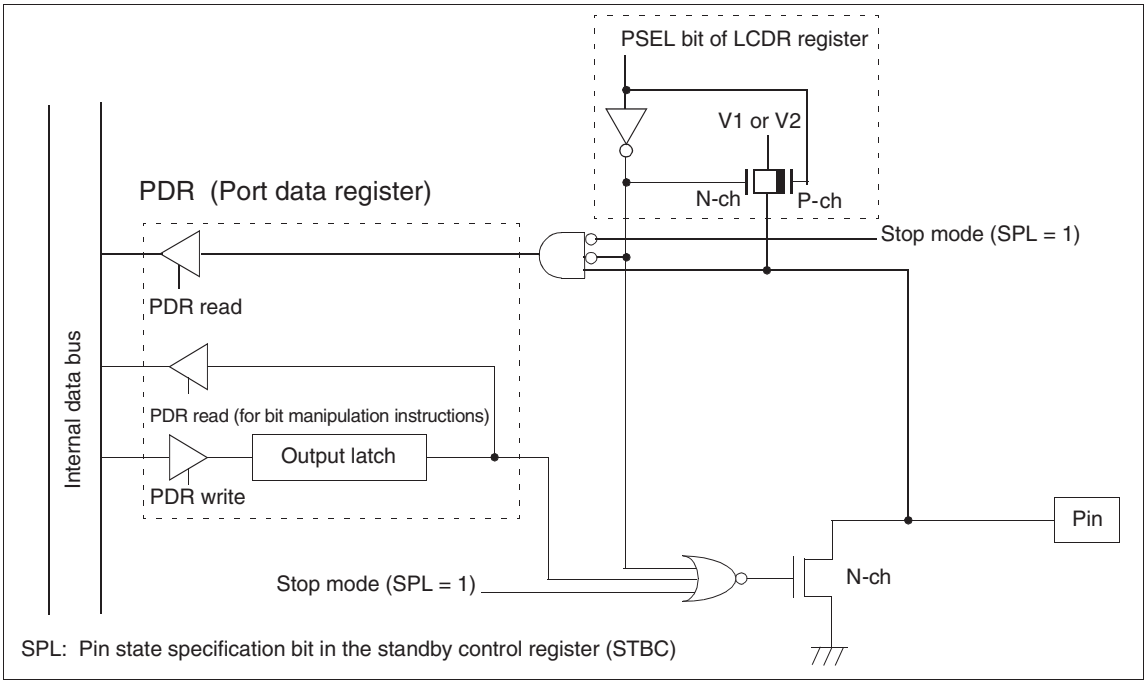


Figure 4.5-2 Block diagram of port 3 pins (P32/V1 and P33/V2)



■ Port 3 register

The port 3 register consists of PDR3. Each bit in the register has a one-to-one relationship with a port 3 pin.

Table 4.5-2 "Correspondence between pin and register for port 3" shows the correspondence between the pins and register for port 3.

Table 4.5-2 Correspondence between pin and register for port 3

Port	Correspondence between register bit and pin								
Port 3	PDR3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	--	--	--	--	P33	P32	P31	P30

4.5.1 Port 3 Data Register (PDR3)

This section describes the port 3 data register.

■ Port 3 data register functions

● Port 3 data register (PDR3)

The PDR3 register holds the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

Reference:

For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

● Settings as an bias input V1 and V2

The PSEL bit in the LCD control register must be cleared to "0" in order to select P32 and P33 as LCD controller bias voltage input. When LCD bias voltage input is selected by using the PSEL bit in the LCD control register, these ports can be used as LCD bias voltage input only.

Table 4.5-3 "Port 3 data register function" lists the functions of the port 3 data register.

Table 4.5-3 Port 3 data register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 3 data register (PDR3)	0	Pin state is the "L" level.	Outputs an "L" level to the pin. (Sets "0" to the output latch and turn the output transistor "ON".)	R/W	$000C_H$	$----1111_B$
	1	Pin state is the "H" level.	Sets the pin to the high-impedance state. (Sets "1" to the output latch and turn the output transistor "OFF".)			

R/W: Readable and writable
-: Unused bit

4.5.2 Operation of Port 3

This section describes the operations of the port 3.

■ Operation of port 3

● Operation as an output port

- Writing data to the PDR3 register stores the data in the output latch. When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and high impedance (Hi-Z) is output from the pin.
- Reading the PDR3 register returns the output latch value.

● Operation as an input port

- Writing "0" to the PDR3 register set the port as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Reading the PDR3 register returns the pin value.
- When V1 and V2 are selected by PSEL bit of the LCD control register, the input data is always as "0".

● Operation as V1 and V2

- When V1 and V2 are selected, set the PDR3 register bits corresponding to V1 and V2 pins to "1" to turn the output transistor "OFF".

● Operation at reset

- At reset, these ports serve as LCD controller/driver bias input. Resetting the CPU initializes the PDR3 register values to "1". This turns "OFF" the output transistor for all pins and all pins are in high-impedance (Hi-Z) state. Since PSEL bit of LCD control register will be reset to "0", P32 and P33 will be configured to V1 and V2 after reset.

● Operation in stop mode

- If P32 and P33 are selected as V1 and V2 and stop mode is entered, the voltage at those pins before entering stop mode will be held. For port output, those pins states in stop mode are controlled by SPL bit in the STBC register.
- The output transistors are forcibly turned "OFF" and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device goes to stop mode.

Table 4.5-4 "Port 3 pin state" lists the port 3 pin states.

Table 4.5-4 Port 3 pin state

Pin name	Normal operation sleep mode stop mode (SPL = "0")	Stop mode (SPL = "1")	Reset
P30 to P33/V2	General-purpose I/O ports/bias input	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

4.6 Port 4

Port 4 is a general-purpose I/O port that also serves as the peripheral signal I/O pins. Individual pin can be switched between the port and resource function. This section principally describes the port functions when operating as a general-purpose I/O port. The section describes the port structure and pins, the pin block diagram, and the port registers for port 4.

■ Structure of port 4

Port 4 consists of the following three components:

- General-purpose I/O port/peripheral I/O pins (P40 to P46/INT0)
- Port 4 data register (PDR4)
- Port 4 data direction register (DDR4)

■ Port 4 pins

Port 4 consists of seven I/O pins of CMOS type input and output. Six of these pins are also used as I/O pins for various peripherals. While they are being used by the peripheral, these pins cannot be used as the general-purpose I/O port.

Table 4.6-1 "Port 4 pins" lists the port 4 pins.

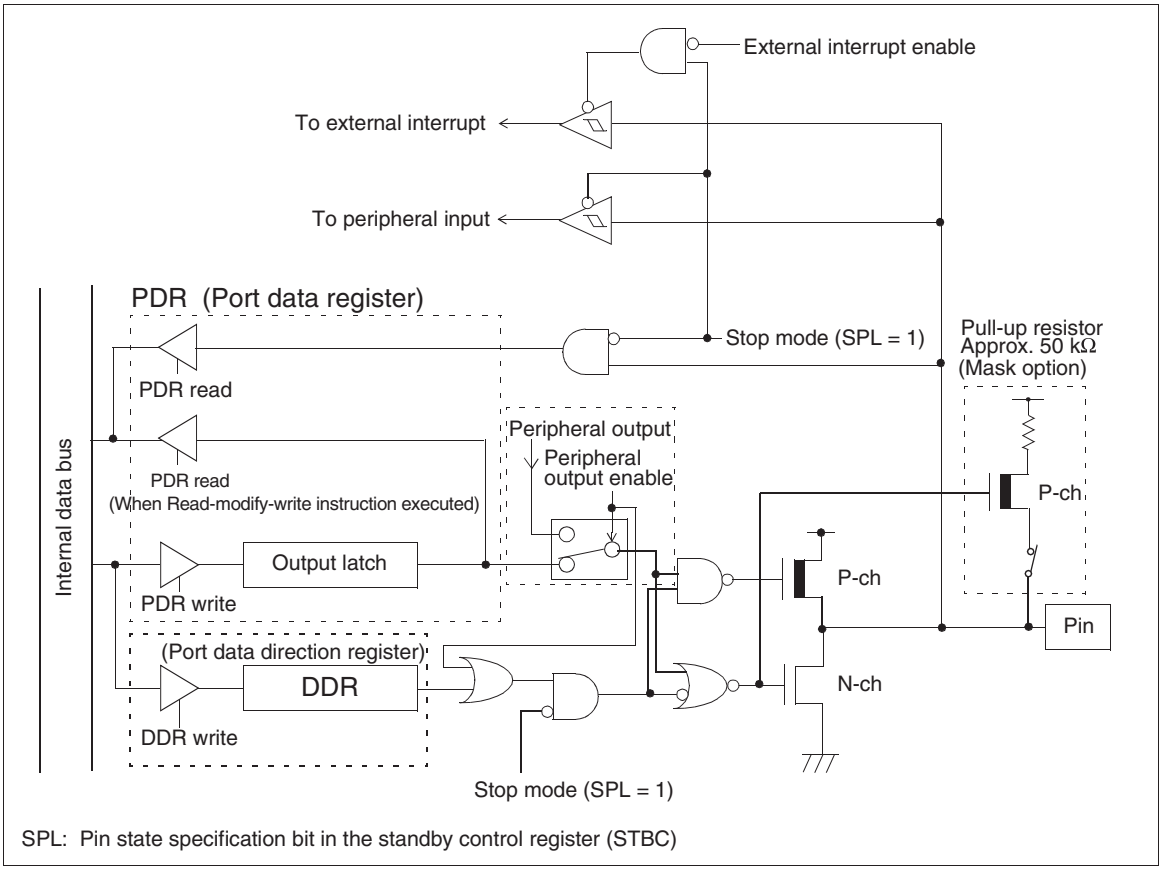
Table 4.6-1 Port 4 pins

Port	Pin name	Function	Shared peripheral	I/O type		Circuit type
				Input	Output	
Port 4	P40	P40 general-purpose I/O port	--	CMOS	CMOS	E
	P41/PWM	P41 general-purpose I/O port	PWM (PWM output)			
	P42/PWC/INT1	P42 general-purpose I/O port	PWC/INT1 (PWC or external interrupt input)			
	P43/SI	P43 general-purpose I/O port	SI (UART/8-bit SIO serial data input)			
	P44/SO	P44 general-purpose I/O port	SO (UART/8-bit SIO serial data output)			
	P45/SCK	P45 general-purpose I/O port	SCK (UART/8-bit SIO serial clock I/O)			
	P46/INT0	P46 general-purpose I/O port	INT0 (External interrupt input)			

See Section 1.7 "I/O Pins and Pin Functions" for a description of the circuit type.

■ Block diagram of port 4 pins

Figure 4.6-1 Block diagram of port 4 pins



SPL: Pin state specification bit in the standby control register (STBC)

Reference:

Peripheral inputs continuously input the pin value (except during stop mode).

■ Port 4 registers

The port 4 registers consist of PDR4 and DDR4.

Each bit in these registers has a one-to-one relationship with a port 4 bit and port 4 pin.

Table 4.6-2 "Correspondence between pin and register for port 4" shows the correspondence between pins and registers for port 4.

Table 4.6-2 Correspondence between pin and register for port 4

Port	Correspondence between register bit and pin								
Port 4	PDR4	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	--	P46	P45	P44	P43	P42	P41	P40
	DDR4	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Corresponding pin	--	P46	P45	P44	P43	P42	P41	P40

4.6.1 Port 4 Registers (PDR4, DDR4)

This section describes the port 4 registers.

■ Port 4 register functions

● Port 4 data register (PDR4)

The PDR4 register holds the pin states. Therefore, when used as an output port that is not a peripheral output, it reads out as the same state ("0" or "1") as that of the output data latch; and when it is an input port, the output latch state cannot be read out.

Reference:

As the bit manipulation instructions (SETB and CLRB) read the output latch data rather than the pin level, the instructions do not change the output latch values for bits other than the bit being set or cleared.

● Port 4 data direction register (DDR4)

The DDR4 register sets the direction (input or output) for each pin (bit).

Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

● Settings as a peripheral output

To use a peripheral that has an output pin, set the peripheral output enable bit for that pin to the "enable" state. As can be seen in the block diagram, the peripheral has precedence over the general-purpose port for use of the output pin. Once the peripheral output is enabled, the states set in the PDR4 and DDR4 registers are no longer valid, and do not affect the data output by the peripheral, or the enabling of the output.

● Settings as a peripheral input

To use a peripheral that has a port 4 pin as an input pin, set that pin as an input port. The output latch data for that pin will no longer be valid.

Table 4.6-3 "Port 4 PDR and DDR register function" lists the functions of the port 4 PDR and DDR registers.

Table 4.6-3 Port 4 PDR and DDR register function

Register	Data	Read	Write	Read/ Write	Address	Initial value
Port 4 data register (PDR4)	0	Pin state is the "L" level.	Outputs an "L" level to the pin if the pin functions as an output port. (Sets "0" to the output latch and turns the output transistor "ON".)	R/W	$000E_H$	$-XXXXXXX_B$
	1	Pin state is the "H" level.	Sets the pin to the high-impedance state if the pin functions as an output port. (*) (Sets "1" to the output latch and turns the output transistor "OFF".)			
Port 4 data direction register (DDR4)	0	--	Disables the output transistor and sets the pin as an input pin.	W	$000F_H$	-0000000_B
	1	--	Enables the output transistor and sets the pin as an output pin.			

R/W: Readable and writable
W: Write-only
X: Indeterminate
-: Unused bit
*: Pins with a pull-up resistor (optional), go to the "H" level

4.6.2 Operation of Port 4

This section describes the operations of the port 4.

■ Operation of port 4

● Operation as an output port

- Setting the corresponding DDR4 register bit to "1" sets a pin as an output port.
- When a pin is as an output port, the output transistor is enabled and the pin outputs the data stored in the output latch.
- Writing data to the PDR4 register stores the data in the output latch and outputs the data to the pin.
- Reading the PDR4 register returns the pin value.

● Operation as an input port

- Setting the corresponding DDR4 register bit to "0" sets a pin as an input port.
- When a pin is set as an input port, the output transistor is "OFF" and the pin goes to the high-impedance state.
- Writing data to the PDP4 register stores the data in the output latch but does not output the data to the pin.
- Reading the PDR4 register returns the pin value.

● Operation as a peripheral output

- If a peripheral output enable bit is set to "enable", the corresponding pin becomes a peripheral output.
- As the pin value can be read even if the peripheral output is enabled, the peripheral output value can be read via the PDR4 register.

● Operation as a peripheral input

- A port pin is set as a peripheral input by setting the corresponding DDR4 register bit to "0".
- Reading the PDR4 register returns the pin value, regardless of whether or not the peripheral is using the input pin.

● Operation at reset

- Resetting the CPU initializes the DDR4 register value to "0". This sets output transistors "OFF" (pins become input ports) and sets the pins to the high-impedance state.
- The PDR4 register is not initialized by a reset. Therefore, to use as output port, the output data must be set in the PDR4 register before setting the corresponding DDR4 register bit to output mode.

● Operation in stop mode

- The pins go to the high-impedance state, if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device goes to stop mode. This is achieved by forcibly setting the output transistor "OFF" regardless of the DDR4 register value.

Table 4.6-4 "Port 4 pin state" lists the port 4 pin states.

Table 4.6-4 Port 4 pin state

Pin name	Normal operation sleep mode stop mode (SPL = "0")	Stop mode (SPL = "1")	Reset
P40 to P46/INT0	General-purpose I/O port/peripheral I/O	Hi-Z	Hi-Z

SPL: Pin state specification bit in the standby control register (STBC)
Hi-Z: High impedance

Reference:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

4.7 Program Example for I/O Ports

This section gives an example program for using the I/O ports.

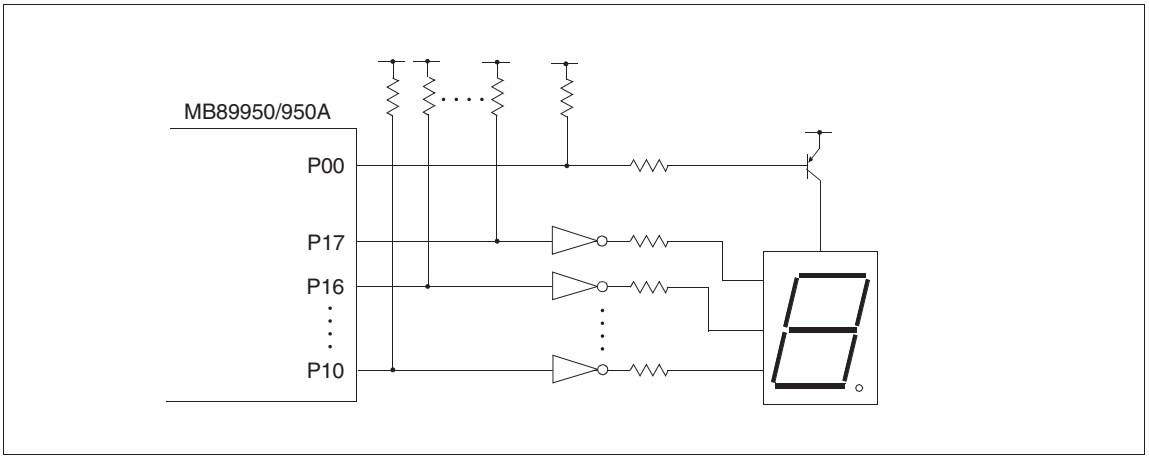
■ Program example for I/O ports

● Processing description

- Port 0 and port 1 are used to illuminate all elements of seven segment LED (eight segments if the decimal point is included).
- The P00 pin is used for the anode common pin of the LED and the P10 to P17 pins operate as the segment pins.

Figure 4.7-1 "Connection example for an eight segment LED" shows the connection example for an eight segment LED.

Figure 4.7-1 Connection example for an eight segment LED



● Coding example

```
PDR0  EQU  0000H              ;Address of the port 0 data register
DDR0  EQU  0001H              ;Address of the port 0 direction register
;---------------Main program-----------------------------------------
      CSEG                    ; [CODE SEGMENT]
       :
      CLRB  PDR0:0            ;Set P00 to the "L" level
      MOV   PDR1, #11111111B  ;Set all port 1 pins to the "H" level
       :
      ENDS
;--------------------------------------------------------------------
      END
```

CHAPTER 5

TIMEBASE TIMER

This chapter describes the functions and operation of the timebase timer.

5.1 "Overview of Timebase Timer"

5.2 "Block Diagram of Timebase Timer"

5.3 "Timebase Timer Control Register (TBTC)"

5.4 "Timebase Timer Interrupt"

5.5 "Operation of Timebase Timer"

5.6 "Notes on Using Timebase Timer"

5.7 "Program Example for Timebase Timer"

5.1 Overview of Timebase Timer

The timebase timer provides interval timer functions. Four different interval times can be selected. The timebase timer uses a 20-bit free-run counter which counts up in synchronous with the internal count clock (divide-by-two the main clock oscillation frequency). The timebase timer also provides the timer output for the oscillation stabilization delay time, the operating clock for the watchdog timer, the operating clock for LCD controller/driver and sampling clock for noise filter circuit in PWC timer.
The timebase timer stops operating in stop mode.

■ Interval timer function

The interval timer function generates repeated interrupts at fixed time intervals.

- The timer generates an interrupt each time the interval timer bit overflows on the timebase timer counter.
- The interval timer bit (interval time) can be selected from four different settings.

Table 5.1-1 "Timebase timer interval time" lists the available interval time for the timebase timer.

Table 5.1-1 Timebase timer interval time

Internal count clock cycle	Interval time (F_{CH} = 5 MHz)
$2/F_{CH}$	$2^{15}/F_{CH}$ (approx. 6.55 ms)
	$2^{17}/F_{CH}$ (approx. 26.21 ms)
	$2^{19}/F_{CH}$ (approx. 104.86 ms)
	$2^{21}/F_{CH}$ (approx. 419.43 ms)

F_{CH}: Main clock oscillation frequency

■ Clock supply function

The clock supply function provides the timer output used for the main clock oscillation stabilization delay time (two values), and operation clock for some peripheral functions.

Table 5.1-2 "Clocks supplied by timebase timer" lists the cycles of the clocks that the timebase timer supplies to various peripherals.

Table 5.1-2 Clocks supplied by timebase timer

Clock destination	Clock cycle	Remarks
Main clock oscillation stabilization delay time	$2^{14}/F_{CH}$ (approx. 3.28 ms)	Selected by mask option
	$2^{18}/F_{CH}$ (approx. 52.43 ms)	
Noise-filter circuit in PWC timer	$2^{2}/F_{CH}$ (approx. 0.8 µs)	Select by noise filter control register
	$2^{5}/F_{CH}$ (approx. 6.4 µs)	
	$2^{7}/F_{CH}$ (approx. 25.6 µs)	
Watchdog timer	$2^{21}/F_{CH}$ (approx. 419.43 ms)	Count-up clock for the watchdog timer
LCD controller/driver	$2^{6}/F_{CH}$ (approx. 12.8 µs)	Frame cycle clock

F_{CH}: Main clock oscillation frequency

The values enclosed in parentheses () are for a 5 MHz main clock oscillation frequency.

Reference:

The oscillation stabilization delay time should be used as a guide line since the oscillation cycle is unstable immediately after oscillation starts.

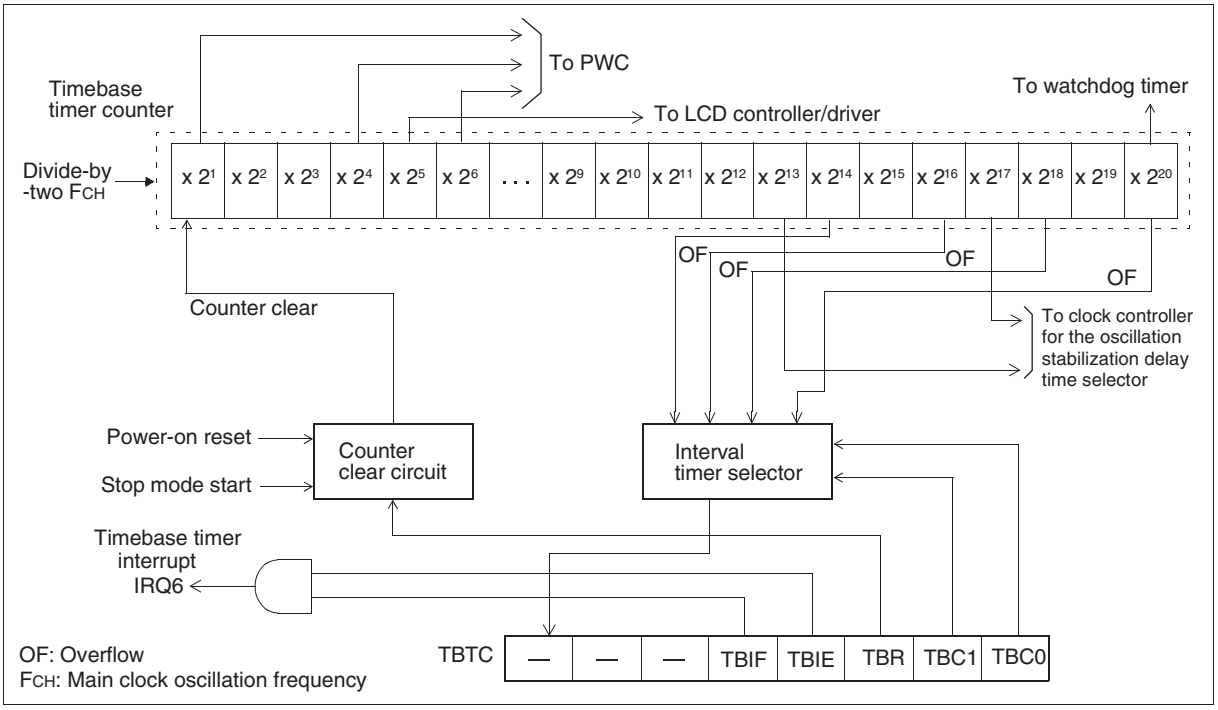
5.2 Block Diagram of Timebase Timer

The timebase timer consists of the following four blocks:

- **Timebase timer counter**
- **Counter clear circuit**
- **Interval timer selector**
- **Timebase timer control register (TBTC)**

■ Block diagram of timebase timer

Figure 5.2-1 Block diagram of timebase timer



● Timebase timer counter

A 20-bit up counter that uses the divide-by-two main clock oscillation frequency as a count clock. The counter stops when the main clock oscillator is stopped.

● Counter clear circuit

In addition to being cleared by setting the TBTC register (TBR = "0"), the counter is cleared when device goes to stop mode (STBC: STP = "1") or by power-on reset (optional).

● Interval timer selector

Selects one of four operating timebase timer counter bits as the interval timer bit. An overflow on the selected bit triggers an interrupt.

● TBTC register

The TBTC register is used to select the interval timer bit, clear the counter, control interrupts, and check the state of the timebase timer.

5.3 Timebase Timer Control Register (TBTC)

The timebase timer control register (TBTC) is used to select the interval times bit, clear the counter, control interrupts, and check the state of the timebase timer.

■ Timebase timer control register (TBTC)

Figure 5.3-1 Timebase timer control register (TBTC)

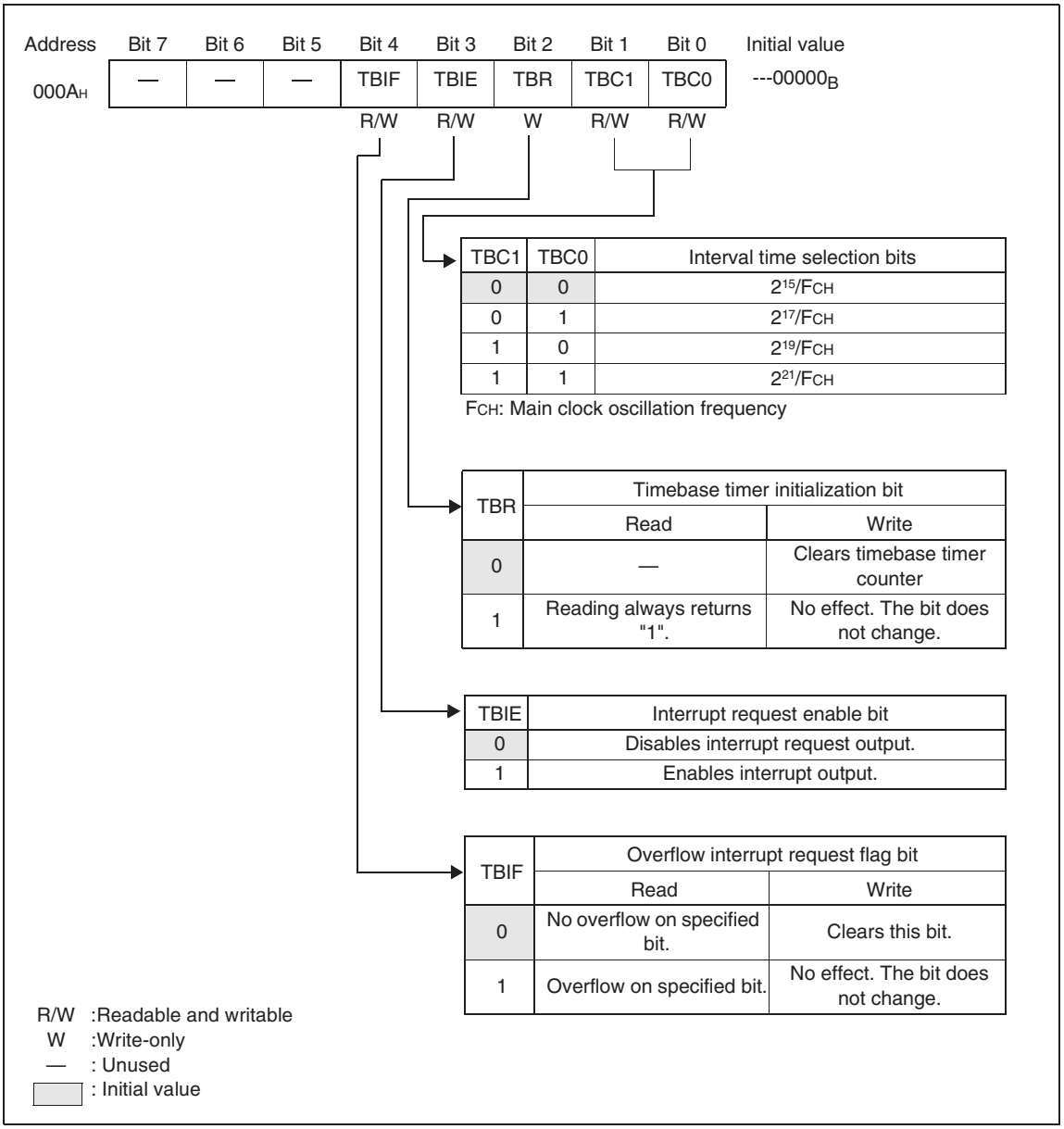


Table 5.3-1 Timebase timer control register (TBTC) bits

Bit		Function
Bit 7 Bit 6 Bit 5	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 4	TBIF: Overflow interrupt request flag bit	• This bit is set to "1" when counter overflow occurs on the specified bit of the timebase timer counter. • An interrupt request is generated when both this bit and the interrupt request enable bit (TBIE) are "1". • Writing "0" clears this bit. • Writing "1" has no effect and does not change the bit value.
Bit 3	TBIE: Interrupt request	• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the overflow interrupt request flag bit (TBOF) are "1".
Bit 2	TBR: Timebase timer initialization bit	• This bit clears the timebase timer counter. • Writing "0" to this bit clears the counter to "00000_H". Writing "1" has no effect and does not change the bit value. **Note:** The read value is always "1".
Bit 1 Bit 0	TBC1, TBC0: Interval time selection bits	• The read value is indeterminate. • These bits select the cycle of the interval timer. • These bits select which bit of the timebase timer counter to use as the interval timer bit. • Four different interval times can be selected.

5.4 Timebase Timer Interrupt

The timebase timer can generate an interrupt request when an overflow occurs on the specified bit of the timebase counter (for the interval timer function).

■ Interrupts for interval timer function

The counter counts up on the internal count clock. When an overflow occurs on the selected interval timer bit, the overflow interrupt request flag bit (TBTC: TBIF) is set to "1". At this time, an interrupt request (IRQ6) to the CPU is generated if the interrupt request enable bit is enabled (TBTC: TBIE = "1"). Write "0" to the TBIF bit in the interrupt processing routine to clear the interrupt request. The TBIF bit is set when at the specified counter bit overflows, regardless of the TBIE bit value.

Note:

When enabling an interrupt request output (TBIE = "1") after wake-up from a reset, always clear the TBIF bit (TBIF = "0") at the same time.

References:

- An interrupt request is generated immediately if the TBIF bit is "1" when the TBIE bit is changed from disabled to enabled ("0" --> "1").
- The TBIF bit is not set if the counter is cleared (TBTC: TBR = "0") at the same time as an overflow on the specified bit occurs.

■ Oscillation stabilization delay time and timebase timer interrupt

If the interval time is set shorter than the main clock oscillation stabilization delay time, an interval interrupt request from the timebase timer (TBTC: TBIF = "1") is generated at the time when the clock mode starts operation. In this case, disable the timebase timer interrupt when entering to a mode in which the main clock oscillation is stopped (stop mode).

■ Register and vector table for timebase timer interrupts

Table 5.4-1 Register and vector table for timebase timer interrupt

Interrupt	Interrupt level settings register			Vector table address	
	Register	Set bit		Upper	Lower
IRQ6	ILR2 ($007D_H$)	L61 (Bit 5)	L60 (Bit 4)	$FFEE_H$	$FFEF_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupt.

5.5 Operation of Timebase Timer

The timebase timer has the interval timer function and the clock supply function for some peripherals.

■ Operation of interval timer function (timebase timer)

Figure 5.5-1 "Interval timer function settings" shows the settings required to operate the interval timer function.

Figure 5.5-1 Interval timer function settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TBTC	—	—	—	TBIF	TBIE	TBR	TBC1	TBC0
				0	1	0	◎	◎

◎ : Used bit
1 : Set “1”
0 : Set “0”

Provided that the main clock is oscillating, the timebase timer counter continues to count up in synchronous with the internal count clock (divide-by-two main clock oscillation frequency).

After being cleared (TBR = "0"), the counter restarts to count up from zero. The timebase timer sets the overflow interrupt request flag bit (TBIF) to "1" when an overflow occurs on the interval timer bit. Consequently, the timebase timer generates interrupt requests at fixed intervals (the selected interval time), based on the time that the counter is cleared.

■ Operation of clock supply function

The timebase timer is also used as a timer to generate the main clock oscillation stabilization delay time. The time from when the timebase timer counter is cleared and starts counting up until an overflow occurs on the oscillation stabilization delay time bit is the oscillation stabilization delay time.

If the timebase timer output is selected by the watchdog timer counter (WDTC: WTE3 to WTE0 = "0101_B"), clearing the timebase timer counter by entering to stop mode (STBC: STP = "1") will also clear the watchdog timer at the same time.

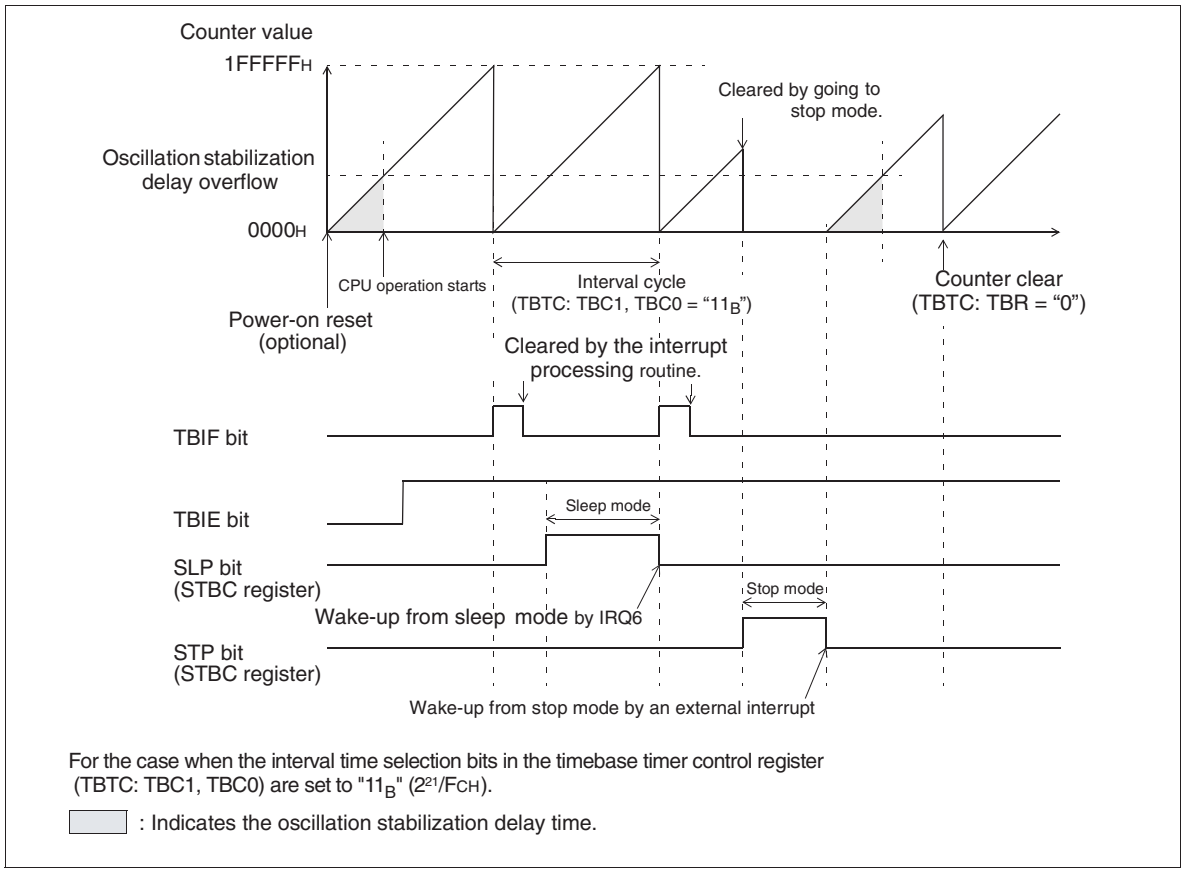
■ Operation of timebase timer

The state of following operations are shown in Figure 5.5-2 "Operation of timebase timer".

- A power-on reset occurs.
- Goes to sleep mode during operation of the interval timer function in the main clock mode.
- Goes to stop mode.
- A counter clear request occurs.

The timebase timer is cleared by going to stop mode, and its operation is stopped. The timebase timer counts the oscillation stabilization delay time after wake-up from stop mode.

Figure 5.5-2 Operation of timebase timer



5.6 Notes on Using Timebase Timer

This section lists points to note when using the timebase timer.

■ Notes on using timebase timer

● Notes on setting bits by program

The system cannot recover from interrupt processing if the overflow interrupt request flag bit (TBTC: TBIF) is "1" and the interrupt request enable bit is enabled (TBTC: TBIE = "1"). Always clear the TBIF bit.

● Clearing timebase timer

In addition to being cleared by the timebase timer initialization bit (TBTC: TBR = "0"), the timer is cleared whenever the main clock oscillation stabilization delay time is required. When the timebase timer is selected as a count clock of the watchdog timer, clearing the timebase timer also clears the watchdog timer.

● Using as timer for oscillation stabilization delay time

As the main clock oscillation frequency is stopped when the power is turned on during stop mode, the timebase timer provides the oscillation stabilization delay time after the oscillator starts.

An appropriate oscillation stabilization delay time must be selected for the type of resonator connected to the main clock oscillator (clock generator).

See Section 3.6.1 "Clock Generator".

● Notes on peripheral functions that provided a clock supply from timebase timer

In modes in which the main clock oscillation frequency is stopped, the timebase timer also stops, and the counter is cleared.

As the clock derived from the timebase timer restarts output from its initial state when the timebase timer counter is cleared, the "H" level may be shorter or the "L" level longer by a maximum of half cycle. The clock of the watchdog timer also restarts output from its initial state.

5.7 Program Example for Timebase Timer

This section gives a program example for the timebase timer.

■ Program example for timebase timer

● Processing description

- Generates repeated interval timer interrupts at $2^{19}/F_{CH}$ (F_{CH}: Main clock oscillation frequency) intervals. At this time, the interval time is approximately 104.86 ms (at 5 MHz operation).

● Coding example

```
TBTC  EQU   0000AH            ; Address of the timebase timer control register

TBIF  EQU   TBTC:3            ; Define the interrupt request flag bit.

ILR2  EQU   007DH             ; Address of the interrupt level setting register 2

INT_V DSEG  ABS               ; [DATA SEGMENT]
      ORG   0FFEEH
IRQ6  DW    WARI              ; Set interrupt vector.
INT_V ENDS
;-----Main program-----------------------------------------------------------
      CSEG                    ; [CODE SEGMENT]
                              ; Stack pointer (SP) etc. are already initialized.
       :
      CLRI                    ; Disable interrupts.
      MOV   ILR2,#11011111B   ; Set interrupt level (level 1).
      MOV   TBTC,#00010010B   ; Clear interrupt request flag, enable interrupt
                                request output, select 2^19/FCH, and clear timebase
                                timer.
      SETI                    ; Enable interrupts.
       :
;-----Interrupt program------------------------------------------------------
WARI  CLRB  TBOF              ; Clear interrupt request flag.
      PUSHW A
      XCHW  A,T
      PUSHW A
       :
      User processing
       :
      POPW  A
      XCHW  A,T
      POPW  A
      RETI
      ENDS
;---------------------------------------------------------------------------
      END
```

CHAPTER 6
WATCHDOG TIMER

This chapter describes the functions and operation of the watchdog timer.

6.1 "Overview of Watchdog Timer"

6.2 "Block Diagram of Watchdog Timer"

6.3 "Watchdog Timer Control Register (WDTC)"

6.4 "Operation of Watchdog Timer"

6.5 "Notes on Using Watchdog Timer"

6.6 "Program Example for Watchdog Timer"

6.1 Overview of Watchdog Timer

The watchdog timer is a 2-bit counter that uses, as its count clock source, the timebase timer derived from the main clock. The watchdog timer resets the CPU if not cleared within a fixed time after activation.

■ Watchdog timer function

The watchdog timer is a counter provided to guard against program runaway. Once activated, the counter must be repeatedly cleared within a fixed time interval. If the program becomes trapped in an endless loop or similar and does not clear the counter within the fixed time, the watchdog timer generates a four-instruction cycle watchdog reset to the CPU.

The timebase timer output can be selected as the watchdog timer count clock.

Table 6.1-1 "Watchdog timer interval time" lists the watchdog timer interval times. If not cleared, the watchdog timer generates a watchdog reset at a time between the minimum and maximum times listed. Clear the counter within the minimum time given in the table.

Table 6.1-1 Watchdog timer interval time

	Count clock
	Timebase timer output (main clock oscillation frequency at 5 MHz)
Minimum time	Approx. 419.43 ms (*1)
Maximum time	Approx. 838.86 ms

*1: Divide-by-two of the main clock oscillation frequency (F_{CH}) x timebase timer count value (2^{20}).

See Section 6.4, "Operation of Watchdog Timer" for the details on the minimum and maximum time of the watchdog timer interval times.

Reference:

The watchdog timer counter is cleared whenever the device goes to sleep or stop mode. Operation halts until the device returns to normal operation (RUN state).

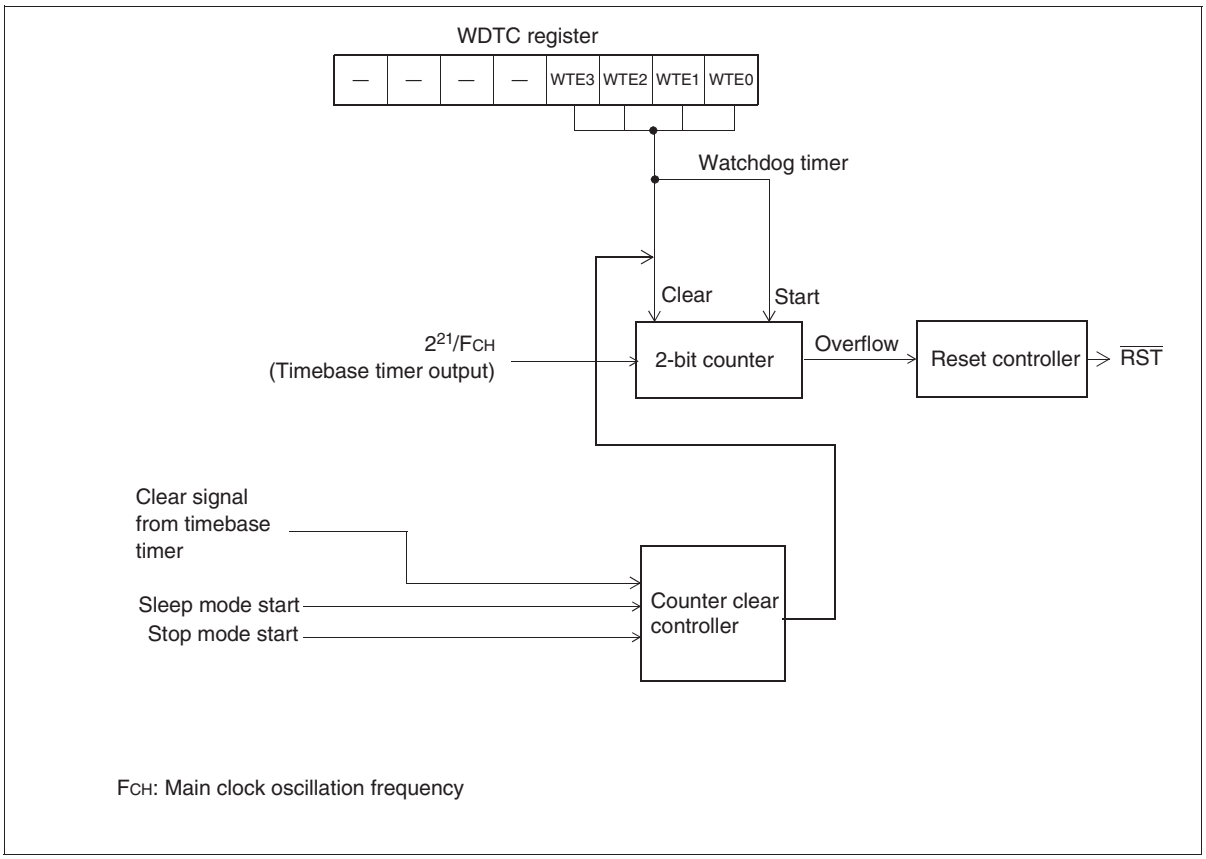
6.2 Block Diagram of Watchdog Timer

The watchdog timer consists of the following four blocks:

- **Watchdog timer counter**
- **Reset controller**
- **Counter clear controller**
- **Watchdog timer control register (WDTC)**

■ Block diagram of watchdog timer

Figure 6.2-1 Block diagram of watchdog timer



● Watchdog timer counter (2-bit counter)

A 2-bit counter that uses the timebase timer output as a count clock.

● Reset controller

Generates a reset signal to the CPU when an overflow occurs on the watchdog timer counter.

● Counter clear controller

Controls clearing and halting the operation of watchdog timer counter.

● WDTC register

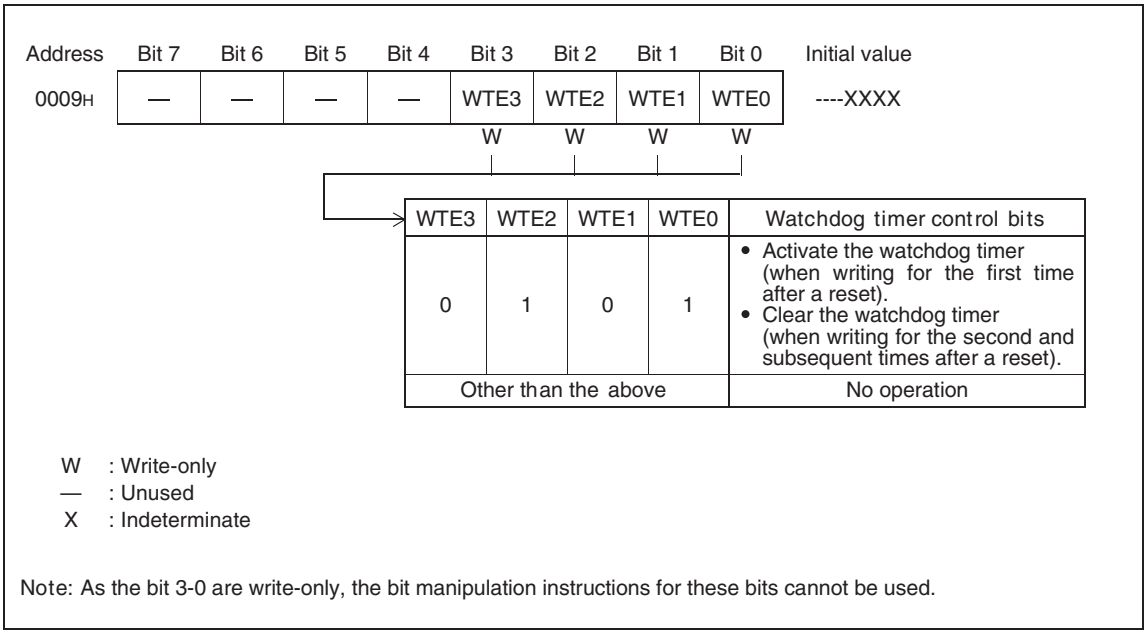
The WDTC register is used to select the count clock, and to activate or clear the watchdog timer counter. As the register is write-only, the bit manipulation instructions cannot be used.

6.3 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) is used to activate or clear the watchdog timer.

■ Watchdog timer control register (WDTC)

Figure 6.3-1 Watchdog timer control register (WDTC)



Note: As the bit 3-0 are write-only, the bit manipulation instructions for these bits cannot be used.

Table 6.3-1 Watchdog timer control register (WDTC) bits

Bit		Function
Bit 7 Bit 6 Bit 5 Bit 4	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on operation.
Bit 3 Bit 2 Bit 1 Bit 0	WTE3, WTE0: Watchdog timer control bits	• Writing "0101_B" to these bits activates (when writing for the first time after a reset) or clears (when writing for the second and subsequent times after a reset) the watchdog timer. • Writing a value other than "0101_B" has no effect on the operation. **Note:** The read value is always "1111_B". The bit manipulation instructions cannot be used.

6.4 Operation of Watchdog Timer

The watchdog timer generates a watchdog reset when the watchdog timer counter overflows.

■ Operation of watchdog timer

● Activating watchdog timer

The watchdog timer is activated by writing "0101_B" to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the first time after a reset.

Once activated, the watchdog timer cannot be stopped other than by a reset.

● Clearing watchdog timer

The watchdog timer counter is cleared by writing "0101_B" to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the second or subsequent times after a reset.

If the counter is not cleared within the interval time of the watchdog timer, the counter overflows and the watchdog timer generates an internal reset signal for four instruction cycles.

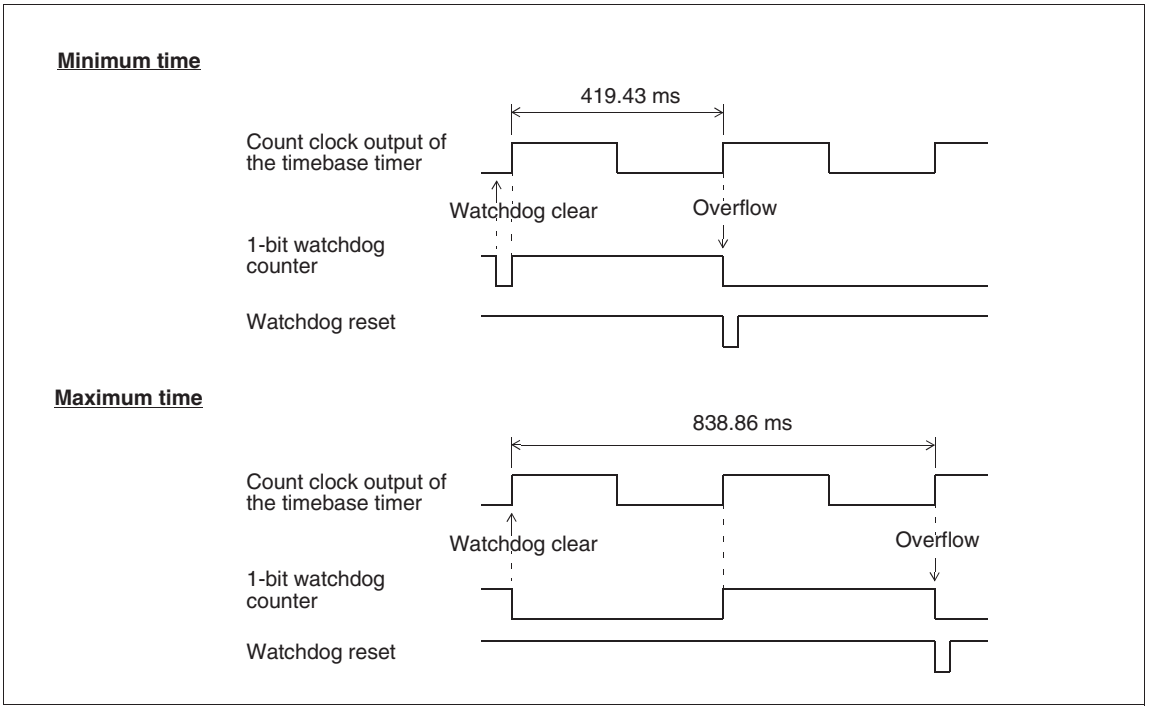
● Interval time of watchdog timer

The interval time changes depending on when the watchdog timer is cleared.

Figure 6.4-1 "Watchdog timer clear and interval time" shows the relationship between the watchdog timer clear timing and the interval time.

The indicated times apply if the timebase timer output is selected as the count clock, and the main clock oscillation frequency is 5 MHz.

Figure 6.4-1 Watchdog timer clear and interval time



6.5 Notes on Using Watchdog Timer

This section lists points to note when using the watchdog timer.

■ Notes on using watchdog timer

● Stopping watchdog timer

Once activated, the watchdog timer cannot stop until a reset generates.

● Clearing watchdog timer

Clearing the counter being used as a count clock of the watchdog timer (timebase timer or watch prescaler) also simultaneously clears the watchdog timer counter.

The watchdog timer counter is cleared when entering sleep or stop mode.

● Notes on programming

When writing a program in which the watchdog timer is repeatedly cleared in the main loop, including interrupt processing, it should be less than the minimum watchdog timer interval time.

6.6 Program Example for Watchdog Timer

This section gives a program example for the watchdog timer.

■ Program example for watchdog timer

● Processing description

Activates the watchdog timer immediately after the program.

Clears the watchdog timer in each loop of the main program.

The processing time for the main loop, including interrupt processing, must be less than the minimum interval time of the watchdog timer (approximately 419.43 ms at 5 MHz operation).

● Coding example

```
WDTC    EQU    0009H          ; Address of the watchdog timer control register
WDT_CLR EQU    00000101B


VECT    DSEG   ABS            ; [DATA SEGMENT]
        ORG    0FFFEH
RST_V   DW     PROG           ; Set reset vector.
VECT ENDS
;-----Main program-----------------------------------------------------------
        CSEG                  ; [CODE SEGMENT]
PROG                          ; Initialization routine after a reset
        MOVW   SP,#0280H      ; Set initial value of stack pointer
                                (for interrupt processing).
         :
        Initialization of peripheral functions (interrupts), etc.
         :
INIT    MOV    WDTC,#WDT_CLR  ; Activate the watchdog timer.

         :
MAIN    MOV    WDTC,#WDT_CLR  ; Clear the watchdog timer.
         :
        User processing (interrupt processing may occur during this cycle)
         :
        JMP    MAIN           ; The loop must be executed in less than the
                                minimum interval time of the watchdog timer.
        ENDS
;---------------------------------------------------------------------------
        END
```

CHAPTER 7

8-BIT PWM TIMER

This chapter describes the functions and operation of the 8-bit PWM timer.

7.1 "Overview of 8-bit PWM Timer"

7.2 "Block Diagram of 8-bit PWM Timer"

7.3 "Structure of 8-bit PWM Timer"

7.4 "8-bit PWM Timer Interrupts"

7.5 "Operation of Interval Timer Function"

7.6 "Operation of PWM Timer Function"

7.7 "States in Each Mode during 8-bit PWM Timer Operation"

7.8 "Notes on Using 8-bit PWM Timer"

7.9 "Program Example for 8-bit PWM Timer"

7.1 Overview of 8-bit PWM Timer

The 8-bit PWM timer can be selected to function as either an interval timer or PWM timer with 8-bit resolution. The interval timer function counts up in synchronous with PWC output clock or one of three internal count clocks. Therefore, an 8-bit interval timer time can be set and the output can be used to generate variable frequency square waves.

■ Interval timer function (square wave output function)

The interval timer function generates repeated interrupts at variable time intervals.

Also, as the 8-bit PWM timer can invert the output level of the pin (PWM) each time an interrupt is generated, the 8-bit PWM timer can output a variable frequency square waves.

- The interval timer can operate with a cycle among 1 and 2^8 times the count clock cycle.
- The count clock can be selected from four different clocks.

Table 7.1-1 "Interval time and square wave output range" lists the range for the interval time and square wave output.

Table 7.1-1 Interval time and square wave output range

	Count clock cycle		Interval time	Square wave output (Hz)
1	Internal count clock	1 t_{inst}	1 t_{inst} to 2^8 t_{inst}	1/(2 t_{inst}) to 1/(2^9 t_{inst})
2		2^4 t_{inst}	2^4 t_{inst} to 2^{12} t_{inst}	1/(2^5 t_{inst}) to 1/(2^{13} t_{inst})
3		2^6 t_{inst}	2^6 t_{inst} to 2^{14} t_{inst}	1/(2^7 t_{inst} to 1/(2^{15} t_{inst})
4	PWC timer output cycle	2 t_{inst} to 2^9 t_{inst}	2 t_{inst} to 2^{17} t_{inst}	1/(2^2 t_{inst}) to 1/(2^{18} t_{inst})
		2^3 t_{inst} to 2^{11} t_{inst}	2^3 t_{inst} to 2^{19} t_{inst}	1/(2^4 t_{inst}) to 1/(2^{20} t_{inst})
		2^6 t_{inst} to 2^{14} t_{inst}	2^6 t_{inst} to 2^{22} t_{inst}	1/(2^7 t_{inst}) to 1/(2^{23} t_{inst})

t_{inst}: Instruction cycle

Reference:

[Calculation example for the interval time and square wave frequency]

In this example, the main clock oscillation frequency (F_{CH}) is 5 MHz, the PWM compare register (COMR) value is set to "DD_H (221)", and the count clock cycle is set to 1 t_{inst}. In this case, the interval time and the frequency of the square wave output from the PWM pin (where the PWM timer operates continuously and the value of the COMR register is constant) are calculated as follows.

Interval time = (1 x 4/FCH) x (COMR register value + 1)
= (4/5 MHz) x (221 + 1)
= 177.6 μs

Output frequency = FCH / (1 x 8 x (COMR register value + 1))
= 5 MHz / (8 x (221 + 1))
= 2.8 kHz

■ PWM timer function

The PWM timer function has 8-bit resolution and can control the "H" and "L" width of one cycle.

- As the resolution is 1/256, pulses can be output with duty ratio of between 0 and 99.6%.
- The cycle of the PWM wave can be selected from four types.
- The PWM timer can be used as a D/A converter by connecting the output to a low-pass filter.

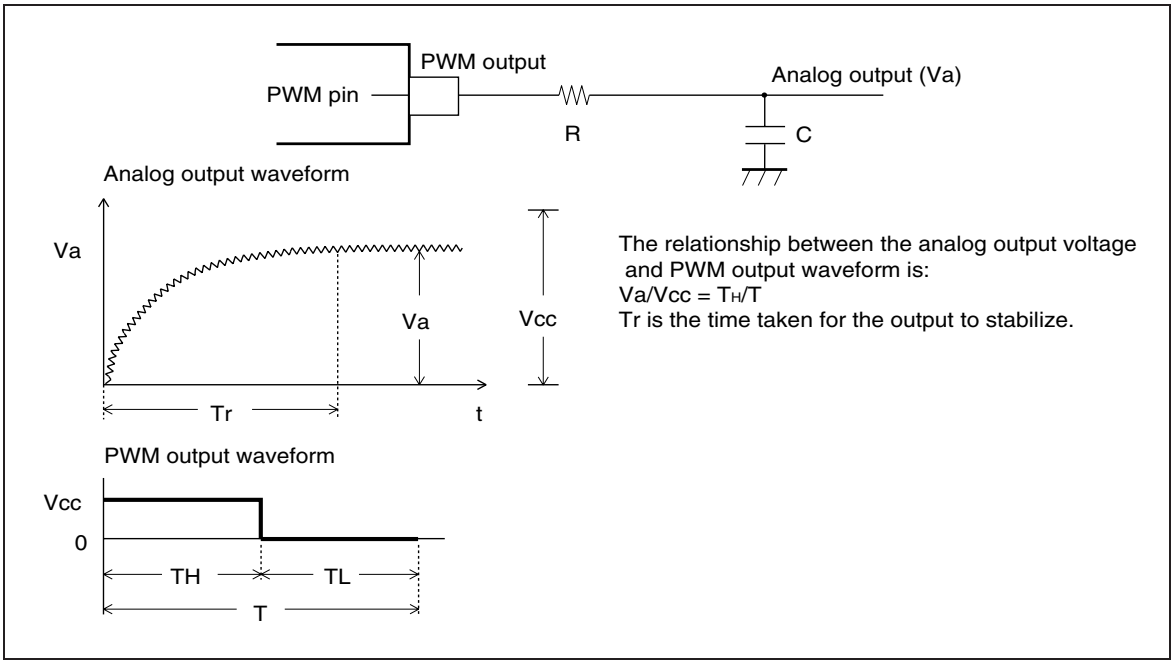
Table 7.1-2 "Available PWM wave cycle for PWM timer function" lists the available PWM wave cycles for the PWM timer function. Figure 7.1-1 "Example D/A converter configuration using PWM output and low-pass filter" shows an example D/A converter configuration.

Table 7.1-2 Available PWM wave cycle for PWM timer function

	1	2	3	4		
	Internal count clock			8-bit timer output cycle times		
Count clock cycle	1 t_{inst}	2^4 t_{inst}	2^6 t_{inst}	2 t_{inst} to 2^9 t_{inst}	2^3 t_{inst} to 2^{11} t_{inst}	2^6 t_{inst} to 2^{14} t_{inst}
PWM wave cycle	2^8 t_{inst}	2^{12} t_{inst}	2^{14} t_{inst}	$2^9$$t_{inst}$ to 2^{17} t_{inst}	2^{11} t_{inst} to 2^{19} t_{inst}	2^{14} t_{inst} to 2^{22} t_{inst}

t_{inst}: Instruction cycle

Figure 7.1-1 Example D/A converter configuration using PWM output and low-pass filter



Reference:

Interrupt requests are not generated during operation of the PWM function.

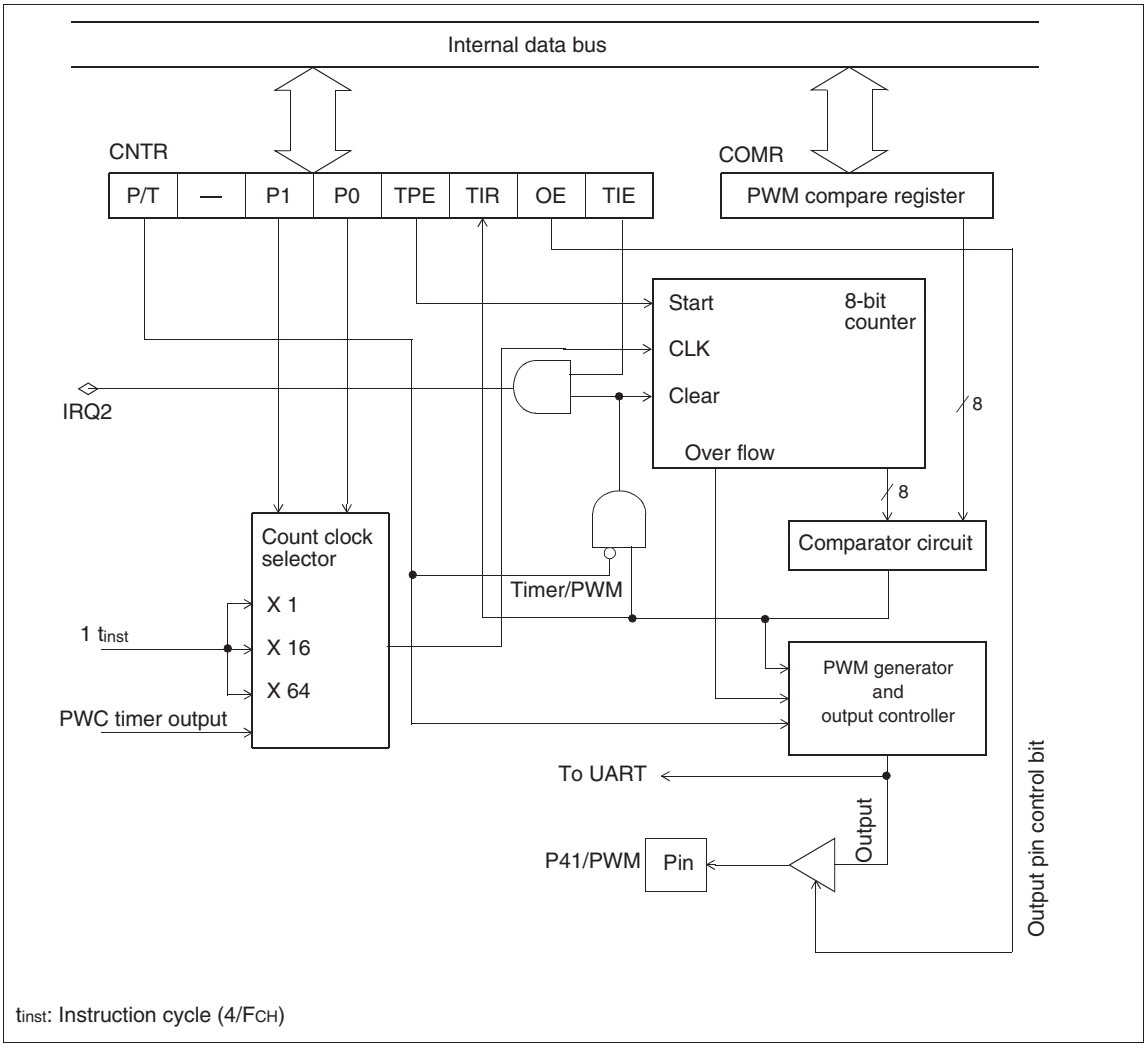
7.2 Block Diagram of 8-bit PWM Timer

The 8-bit PWM timer consists of the following six blocks:

- **Count clock selector**
- **8-bit counter**
- **Comparator circuit**
- **PWM generator and output controller**
- **PWM compare register (COMR)**
- **PWM control register (CNTR)**

■ Block diagram of 8-bit PWM timer

Figure 7.2-1 Block diagram of 8-bit PWM timer



● Count clock selector

Selects a count-up clock for the 8-bit counter from the three internal count clocks and the PWC timer output cycle.

● 8-bit counter

The 8-bit counter counts up on the count clock selected by the count clock selector.

● Comparator circuit

The comparator circuit has a latch to hold the COMR register value. The circuit latches the COMR register value when the 8-bit counter value is "00_H". The comparator circuit compares the 8-bit counter value with the latched COMR register value, and detects when a match occurs.

● PWM generator and output controller

When a match is detected during interval timer operation, an interrupt request is generated and, if the output pin control bit (CNTR: OE) is "1", the output controller inverts the output level of the PWM pin. At the same time, the 8-bit counter is cleared.

When a match is detected during PWM timer operation, the PWM generator changes the output level of the PWM pin from "H" to "L". The pin is set back to the "H" level when the next overflow occurs on the 8-bit counter.

● COMR register

The COMR register is used to set the value that is compared with the value of the 8-bit counter.

● CNTR register

The CNTR register is used to select the operating mode, enable or disable operation, set the count clock, control interrupts, and check the PWM status.

Setting the operation to PWM timer mode (P/T = "0") disables clearing of the 8-bit counter and generation of interrupt requests (IRQ2) when the comparator circuit detects a match.

7.3 Structure of 8-bit PWM Timer

This section describes the pin, pin block diagram, register source, and interrupts of the 8-bit PWM timer.

■ 8-bit PWM timer pin

The 8-bit PWM timer uses the P41/PWM pin. This pin can function as a CMOS general-purpose I/O port (P41), or as the interval timer or PWM timer output (PWM).

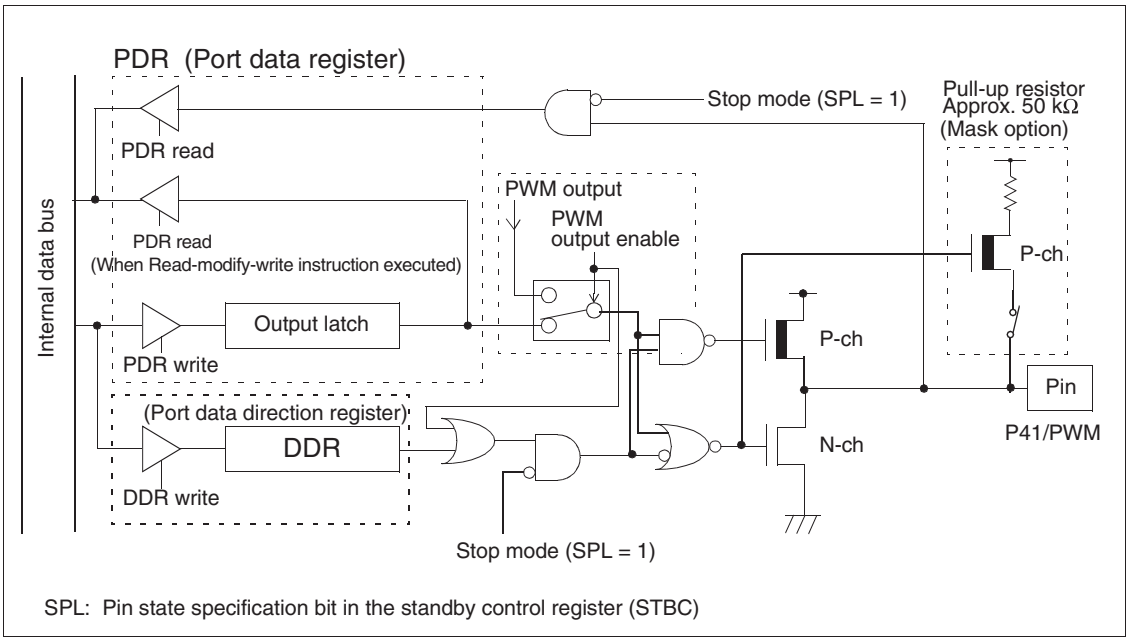
PWM: When the interval timer function is selected, the square waves are output to this pin.

When the PWM timer function is selected, the pin outputs the PWM wave.

Setting the output pin control bit (CNTR: OE) to "1" makes pin P41/PWM the output-only pin for 8-bit PWM timer. Once this has been done, the pin performs its PWM function regardless of the state of the port data register output latch data (PDR4: bit 1).

■ Block diagram of 8-bit PWM timer pin

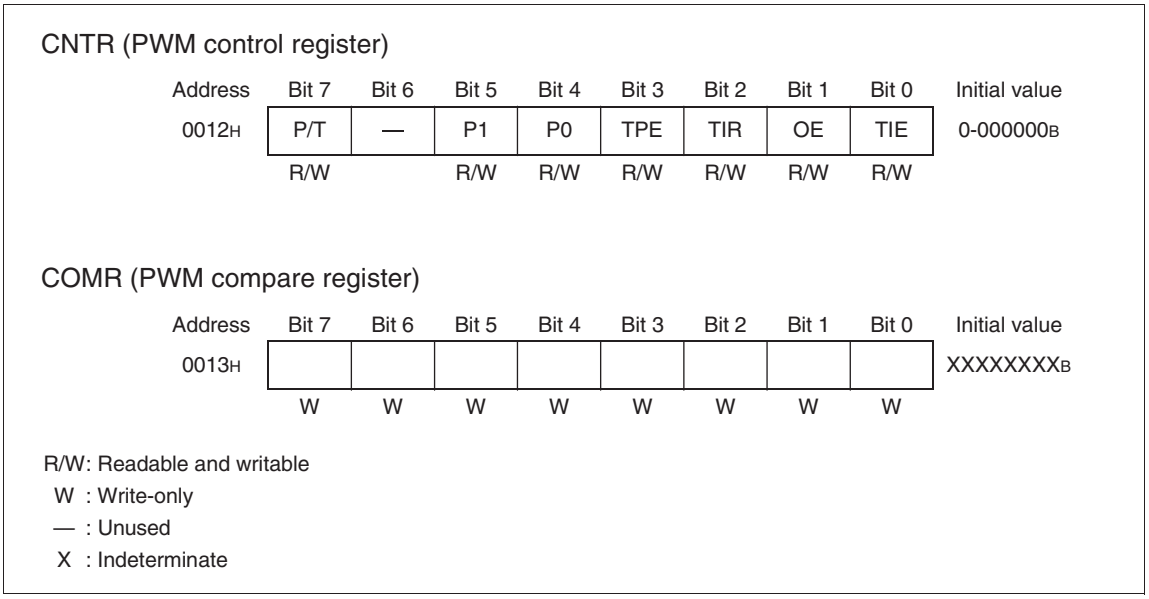
Figure 7.3-1 Block diagram of 8-bit PWM timer pin



SPL: Pin state specification bit in the standby control register (STBC)

■ 8-bit PWM timer registers

Figure 7.3-2 8-bit PWM timer registers



Note:

As the PWM compare register (COMR) is write-only, the bit manipulation instructions cannot be used.

■ 8-bit PWM timer interrupt source

IRQ2:

For the interval timer function, the 8-bit PWM timer generates an interrupt request if interrupt request output is enabled (CNTR: TIE = "1") when the counter value matches the value set in the COMR register. (For PWM function, no interrupt request is generated.)

7.3.1 PWM Control Register (CNTR)

The PWM control register (CNTR) is used to select the operating mode of the 8-bit PWM timer (interval timer operation or PWM timer operation), enable or disable operation, select the count clock, control interrupts, and check the state of the 8-bit PWM timer.

■ PWM control register (CNTR)

Figure 7.3-3 PWM control register (CNTR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0012H	P/T	—	P1	P0	TPE	TIR	OE	TIE	0-000000B
	R/W		R/W	R/W	R/W	R/W	R/W	R/W	

TIE	Interrupt request enable bit
0	Disables interrupt request output.
1	Enables interrupt request output.

OE	Output pin control bit
0	Functions as a general-purpose port (P41).
1	Functions as the interval timer/PWM timer output pin (PWM).

TIR	Interrupt request flag bit		
	Read		Write
	Interval timer function	PWM timer function	
0	Counter value and set value do not match.	No change	Clears this bit.
1	Counter value and set value match.		No effect. The bit does not change.

TPE	Counter operation enable bit
0	Stops count operation.
1	Starts count operation.

P1	P0	Clock selection bits		
0	0	Internal count clock	1 t_{inst}	(*1)
0	1		16 t_{inst}	(*1)
1	0		64 t_{inst}	(*1)
1	1	PWC timer output		(*2)

P/T	Operating mode selection bit
0	Operates as an interval timer.
1	Operates as a PWM timer.

*1: t_{inst} : Instruction cycle
*2: The "PWC timer output" referred to here is the PWC timer output cycle

R/W : Readable and writable
— : Unused
□ : Initial value

Table 7.3-1 PWM control register (CNTR) bits

Bit		Function
Bit 7	P/T: Operating mode selection bit	• This bit switches between the interval timer function (P/T = "0") and PWM timer function (P/T = "1"). **Note:** Write to this bit when the counter operation is stopped (TPE = "0"), interrupts are disabled (TIE = "0"), and the interrupt request flag bit is cleared (TIR = "0").
Bit 6	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 5 Bit 4	P1, P0: Clock selection bits	• These bits select the count clock for the interval timer function and PWM timer function. • These bits can select the count clock from three internal count clocks or the output cycle of the PWC timer. **Note:** Do not change P1 and P0 when the counter is operating (TPE = "1").
Bit 3	TPE: Counter operation enable bit	• This bit activates or stops operation of the PWM timer function and interval timer function. • Writing "1" to this bit starts the counter operation. Writing "0" to this bit stops the count and clears the counter to "00_H".
Bit 2	TIR: Interrupt request flag bit	• For the interval timer function: This bit is set to "1" when the counter and PWM compare register (COMR) value match. An interrupt request is issued to the CPU when both this bit and the interrupt request enable bit (TIE) are "1". • For the PWM timer function: Interrupt requests are not generated. • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 1	OE: Output pin control bit	• The P41/PWM pin functions as a general-purpose I/O port (P41) when this bit is set to "0", and a dedicated pin (PWM) when this bit is set to "1". • The PWM pin outputs a square wave when the interval timer function is selected and a PWM waveform when the PWM timer function is selected.
Bit 0	TIE: Interrupt request enable bit	• This bit enables or disables interrupt request output to the CPU. Interrupt request is generated when both this bit and the interrupt request flag bit (TIR) are "1".

7.3.2 PWM Compare Register (COMR)

The PWM compare register (COMR) sets the interval time for the interval timer function. The register value sets the "H" width of the pulse for the PWM timer function.

■ PWM compare register (COMR)

Figure 7.3-4 "PWM compare register (COMR)" shows the bit structure of the PWM compare register.

As the register is write-only, bit manipulation instructions cannot be used.

Figure 7.3-4 PWM compare register (COMR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0013H									XXXXXXXXB
	W	W	W	W	W	W	W	W	

W: Write-only
X: Indeterminate

● Interval timer operation

This register is used to set the value to be compared with the counter value. The register specifies the interval time.

The counter is cleared when the counter value matches the value set in this register, and the interrupt request flag bit is set to "1" (CNTR: TIR = "1").

If data is written to the COMR register during counter operation, the new value applies from the next cycle (after the next match is detected).

Reference:

The COMR setting for interval timer operation can be calculated using the following formula.

COMR register value = interval time/(count clock cycle x instruction cycle) – 1

● PWM timer operation

This register is used to set the value to be compared with the counter value. The register therefore sets the "H" width of the pulse.

The PWM pin outputs an "H" level until the counter value matches the value set in this register. From the match until the counter value overflows, the PWM pin outputs an "L" level.

If data is written to the COMR register during counter operation, the new value applies from the next cycle (after the next overflow).

Reference:

In PWM timer operation, the COMR setting and the PWM cycle time can be calculated using the following formulas.

COMR register value = duty ratio (%) x 256

PWM wave cycle = count clock cycle x instruction cycle x 256

7.4 8-bit PWM Timer Interrupts

The 8-bit PWM timer can generate an interrupt request when a match is detected between the counter value and PWM compare register value for the interval timer function. Interrupt requests are not generated for the PWM timer function.
8-bit PWM timer generates the IRQ2 as an interrupt request.

■ Interrupts for interval timer function

The counter starts to count up from "00_H" on the selected count clock. When the counter value matches the PWM compare register (COMR) value, the interrupt request flag bit (CNTR: TIR) is set to "1".

At this time, an interrupt request (IRQ2) to the CPU is generated if the interrupt request enable bit is enabled (CNTR: TIE = "1"). Write "0" to the TIR bit in the interrupt processing routine to clear the interrupt request.

The TIR bit is set to "1" when the counter value matches the set value, regardless of the value of the TIE bit.

Reference:

The TIR bit is not set if the counter is stopped (CNTR: TPE = "0") at the same time as the counter value matches the COMR register value.

An interrupt request is generated immediately if the TIR bit is "1" when the TIE bit is changed from disabled to enabled ("0" --> "1").

■ Registers and vector tables for 8-bit PWM timer interrupts

Table 7.4-1 Registers and vector tables for 8-bit PWM timer interrupts

	Interrupt	Interrupt level setting register			Vector table address	
		Register	Setting bits		Upper	Lower
8-bit PWM timer	IRQ2	ILR1 ($007C_H$)	L21 (Bit 5)	L20 (Bit 4)	$FFF6_H$	$FFF7_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupt operation.

7.5 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the 8-bit PWM timer.

■ Operation of interval timer function

Figure 7.5-1 "Interval timer function settings" shows the settings required to operate as an interval timer function.

Figure 7.5-1 Interval timer function settings

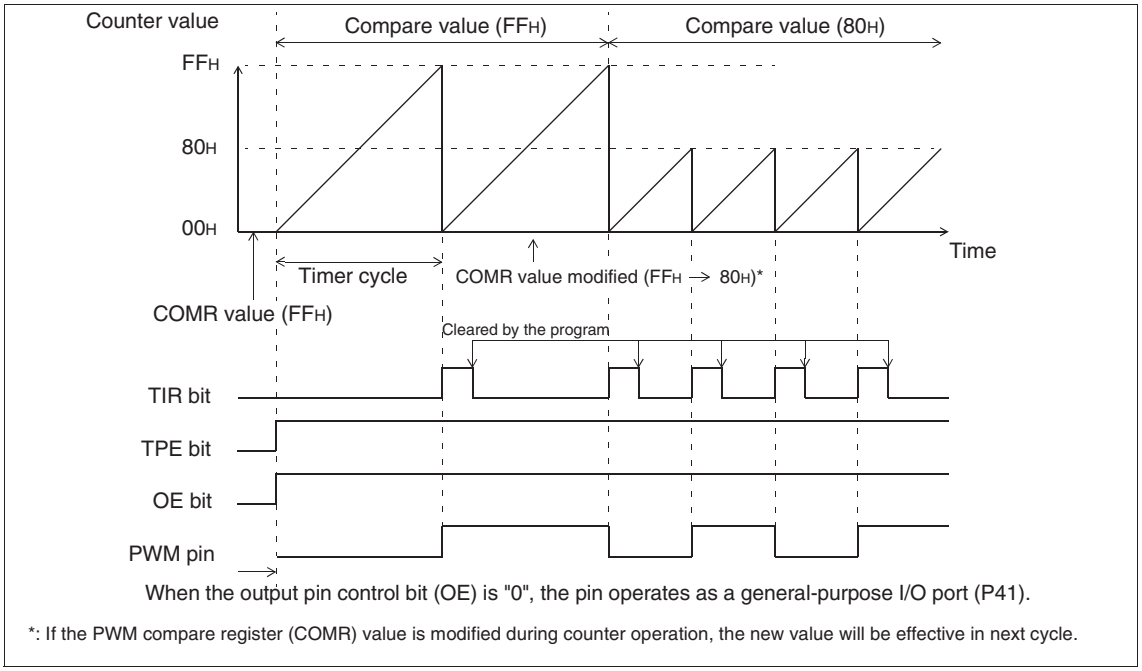
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CNTR	P/T	—	P1	P0	TPE	TIR	OE	TIE
	0		◎	◎	1	◎	◎	◎
COMR	Sets interval time (compare value).							

◎: Used bit
1 : Set "1".
0 : Set "0".

On activation, the counter starts to count up from "00_H" on the rising edge of the selected count clock. When the counter value matches the value set in the COMR register (compare value), the PWM timer inverts the level of the output pin (PWM) on the next rising edge of the count clock, clears the counter, sets the interrupt request flag bit (CNTR: TIR = "1"), and restarts counting from "00_H".

Figure 7.5-2 "Operation of 8-bit PWM timer" shows the operation of the 8-bit PWM timer.

Figure 7.5-2 Operation of 8-bit PWM timer



*: If the PWM compare register (COMR) value is modified during counter operation, the new value will be effective in next cycle.

Note:

Do not change the count clock cycle (CNTR: P1, P0) during operation of the interval timer function (CNTR: TPE = "1").

References:

- Setting the COMR register value to "00_H" causes the PWM pin output to be inverted with the cycle of the selected count clock.
- When the counter is stopped (CNTR: TPE = "0") while the interval timer function is selected, the PWM pin outputs an "L" level.

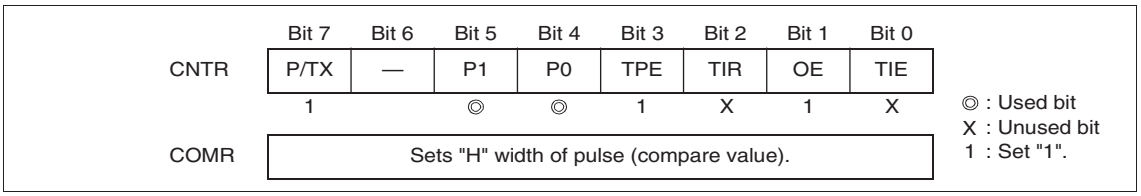
7.6 Operation of PWM Timer Function

This section describes the operation of the PWM timer function of the 8-bit PWM timer.

■ Operation of PWM timer function

Figure 7.6-1 "PWM timer function settings" shows the settings required to operate as the PWM timer function.

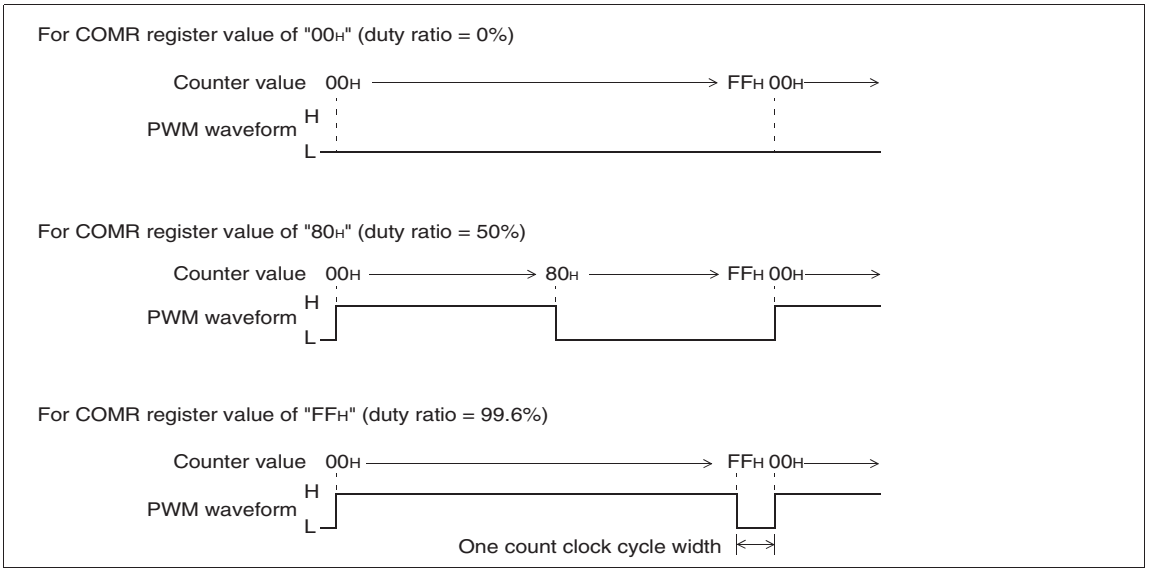
Figure 7.6-1 PWM timer function settings



On activation, the counter starts to count up from "00_H" on the rising edge of the selected count clock. The PWM pin (PWM) outputs (PWM waveform) an "H" level until the counter value matches the value set in the COMR register. From the match until the counter value overflows (FF_H --> 00_H), the PWM pin outputs an "L" level.

Figure 7.6-2 "Example of PWM waveform output (PWM pin)" shows the PWM waveforms output from the PWM pin.

Figure 7.6-2 Example of PWM waveform output (PWM pin)



Note:

Do not change the count clock cycle (CNTR: P1, P0) during operation of the PWM timer function (CNTR: TPE = "1").

Reference:

When the PWM timer function is selected, the PWM pin maintains its existing level when the counter is stopped (CNTR: TPE = "0").

7.7 States in Each Mode during 8-bit PWM Timer Operation

This section describes the operation of the 8-bit PWM timer when the device goes to sleep or stop mode, or an operation halt request occurs during operation.

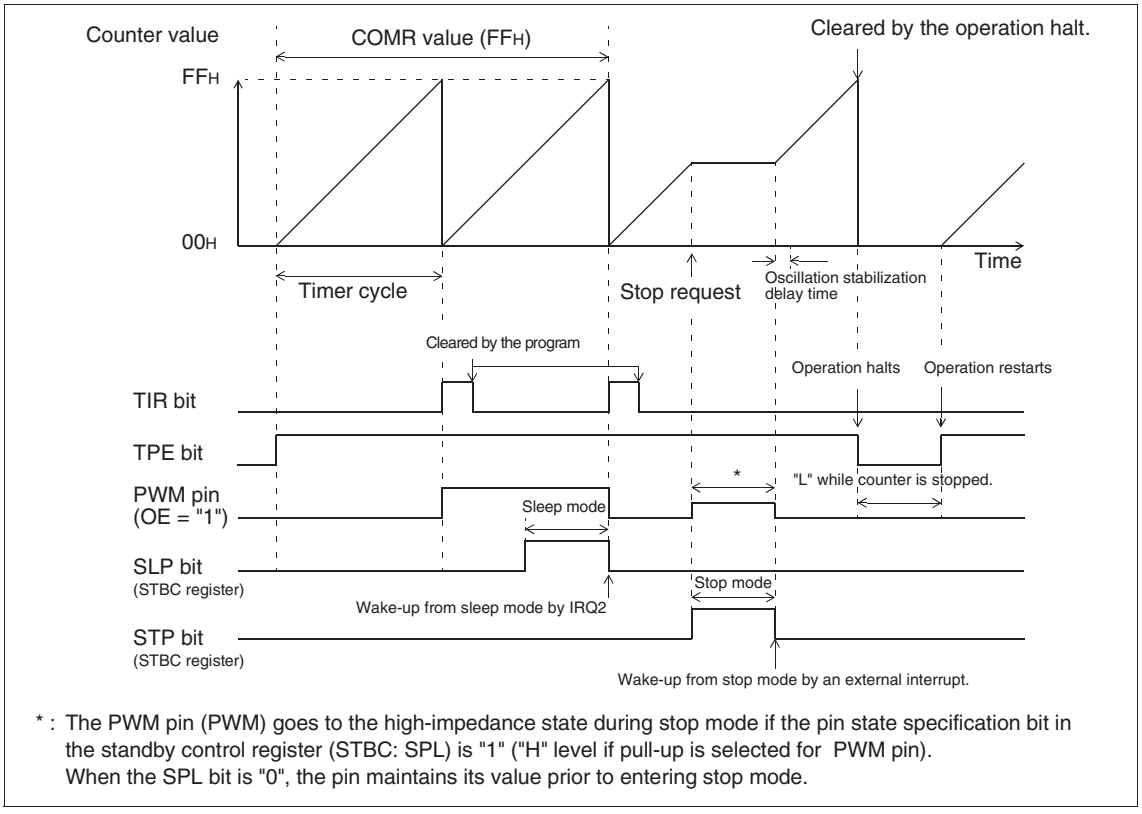
■ Operation during standby mode or operation halt

Figure 7.7-1 "Counter operation during standby mode or operation halt (for interval timer function)" and Figure 7.7-2 "Operation during standby mode or operation halt (for PWM timer function)" show the counter value states when the device goes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or PWM timer function.

The counter halts and maintains its current value when the device goes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. Therefore, the first interval time or PWM wave cycle does not match the set value. Always initialize the 8-bit PWM timer after wake-up from stop mode.

● For interval timer function

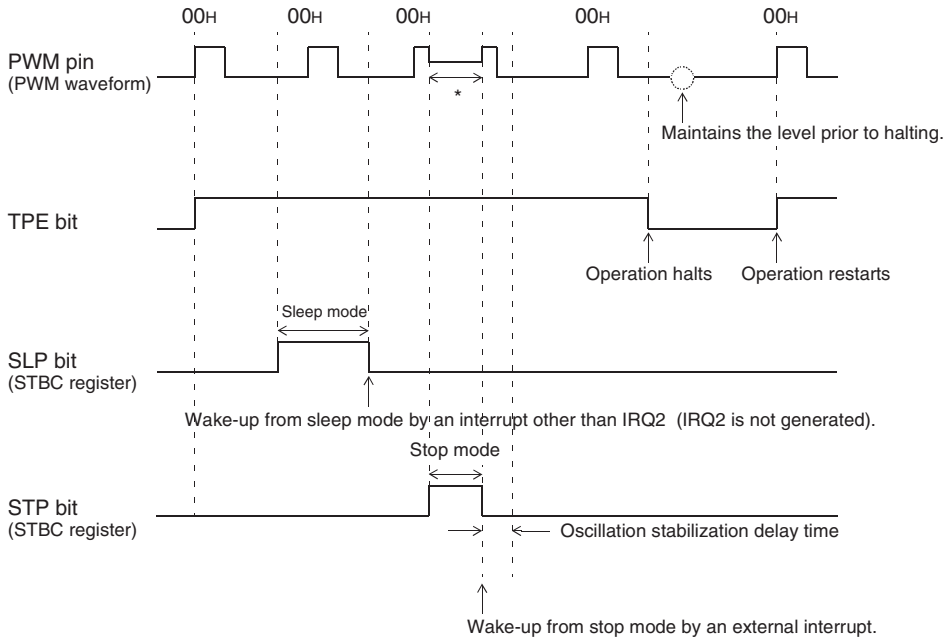
Figure 7.7-1 Counter operation during standby mode or operation halt (for interval timer function)



* : The PWM pin (PWM) goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is "1" ("H" level if pull-up is selected for PWM pin).
When the SPL bit is "0", the pin maintains its value prior to entering stop mode.

● For PWM timer function

Figure 7.7-2 Operation during standby mode or operation halt (for PWM timer function)



* : The PWM pin (PWM) goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is "1" ("H" level if pull-up is selected for PWM pin).
When the SPL bit is "0", the pin maintains its value prior to entering stop mode.

7.8 Notes on Using 8-bit PWM Timer

This section lists points to note when using the 8-bit PWM timer.

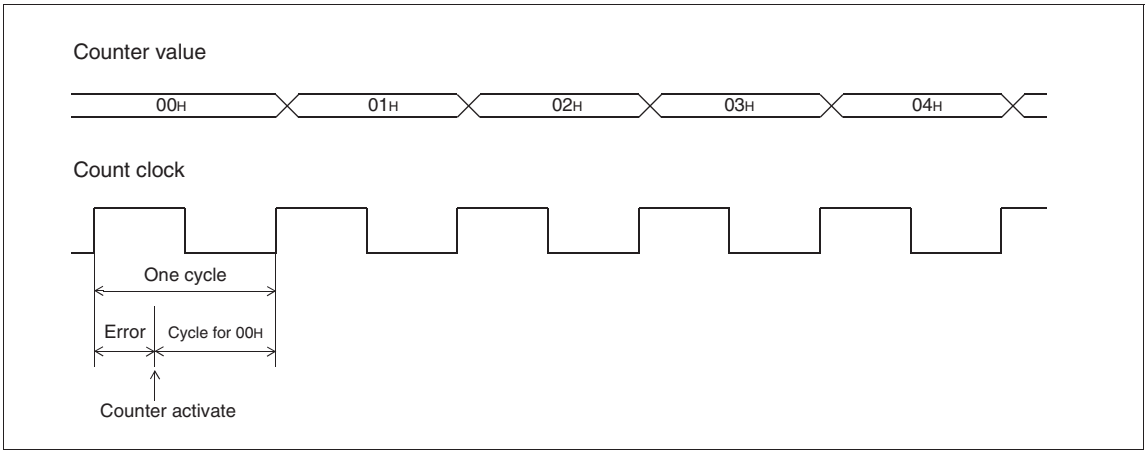
■ Notes on using 8-bit PWM timer

● Error

Activating the counter by program is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter until a match with the PWM compare register (COMR) is detected may be shorter than the theoretical time by a maximum of one cycle count clock.

Figure 7.8-1 "Error on starting counter operation" shows the error that occurs on starting counter operation.

Figure 7.8-1 Error on starting counter operation



● Notes on setting by program

- Do not change the count clock cycle (CNTR: P1, P0) when the interval timer function or PWM timer function is operating (CNTR: TPE = "1").
- Stop the counter (CNTR: TPE = "0"), disable interrupts (TIE = "0") and clear the interrupt request flag (TIR = "0") before switching between the interval timer function and PWM timer function (CNTR: P/T).
- Interrupt processing cannot return if the interrupt request flag bit (CNTR: TIR) is "1" and the interrupt request enable bit is enabled (CNTR: TIE = "1"). Always clear the TIR bit.
- The TIR bit is not set if the counter is disabled (TPE = "0") at the same time as the counter and COMR register values match.

7.9 Program Example for 8-bit PWM Timer

This section gives program examples for the 8-bit PWM timer.

■ Program example for interval timer function

● Processing description

- Generates repeated interval timer interrupts at 2.5 ms intervals.
- Outputs a square wave to the PWM pin that inverts after each interval time.
- With a main clock oscillation frequency F_{CH} of 5 MHz, and the highest speed clock selected by the speed-shift function (1 instruction cycle time = $4/F_{CH}$), the COMR register is set for an interval time of approximately 5 ms (an internal clock period of 64 t_{inst} is selected as the count clock). The COMR register setting is calculated as follows:

 COMR register value = 5 ms/(64 x 4/5 MHz) - 1 = 97 (061_H)

● Coding example

```
CNTR    EQU    0012H           ; Address of the PWM control register
COMR    EQU    0013H           ; Address of the PWM compare register

TPE     EQU    CNTR:3          ; Define the counter operation enable bit.
TIR     EQU    CNTR:2          ; Define the interrupt request flag bit.

ILR1    EQU    007CH           ; Address of the interrupt level setting register 3

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF6H
IRQ2    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   TPE             ; Stop counter operation.
        MOV    ILR1,#11011111B ; Set interrupt level (level 1).
        MOV    COMR,#061H      ; Value compared with the counter value
                                 (interval time)
        MOV    CNTR,#00101011B ; Operate interval timer, select 64 tinst,
                                 start counter operation, clear interrupt request
                                 flag, enable TO pin output, enable interrupt
                                 request output.
        SETI                   ; Enable interrupts.
         :
;-----Interrupt program--------------------------------------------------------
WARI    CLRB   TIR             ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T             ; Save A and T.
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T             ; Restore A and T.
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

■ Program example for PWM timer function

● Processing description

- Generates a PWM wave with a duty ratio of 50%. Then, changes the duty ratio to 25%.
- Does not generate interrupts.
- For a 5 MHz main clock oscillation frequency (F_{CH}), selecting the interval 16 t_{inst} count clock gives a PWM wave cycle of 16 x 4/5 MHz x 256 = 3.277 ms.
- The following shows the COMR register value required for a duty ratio of 50%:

 COMR register value = 50/100 x 256 = 128 (080_H)

● Coding example

```
CNTR    EQU    0012H           ; Address of the PWM control register
COMR    EQU    0013H           ; Address of the PWM compare register

TPE     EQU    CNTR:3          ; Define the counter operation enable bit.
;-----Main program------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
         :
        CLRB   TPE             ; Stop counter operation.
        MOV    COMR,#80H       ; Set "H" width of pulse. Duty ratio = 50%
        MOV    CNTR,#10011010B ; Operate PWM timer, select 16 tinst,
                                 start counter operation, clear interrupt request
                                 flag, enable PWM pin output, and disable
                                 interrupt request output.
         :
         :
        MOV    COMR,#40H       ; Change the duty ratio to 25% (effective from
                                 the next PWM wave cycle).
         :
        ENDS
;----------------------------------------------------------------------------
        END
```

CHAPTER 8

PULSE WIDTH COUNT TIMER (PWC)

This chapter describes the functions and operation of the pulse width count timer (PWC).

8.1 "Overview of Pulse Width Count Timer"

8.2 "Block Diagram of Pulse Width Count Timer"

8.3 "Structure of Pulse Width Count Timer

8.4 "Pulse Width Count Timer Interrupts"

8.5 "Operation of Interval Timer Function"

8.6 "Operation of Pulse Width Measurement Function"

8.7 "Operation of Noise Filter Circuit"

8.8 "States in Each Mode during Pulse Width Count Timer Operation"

8.9 "Notes on Using Pulse Width Count Timer"

8.10 "Program Example for Timer Function of Pulse Width Count Timer"

8.1 Overview of Pulse Width Count Timer

The pulse width count timer (PWC) can be selected to function as either an interval timer or the pulse width measurement. The interval timer function counts down in synchronous with one of three internal count clocks. The pulse width measurement function measures the width of pulses input to an external pin.
Therefore, the PWC can be used as an input capture by continuously measuring the pulse width of an external input.

■ Interval timer function

The interval timer function generates repeated interrupts at variable time intervals.

- The interval timer can operate with a cycle among 1 and 2^8 times the internal count clock cycle.
- The internal count clock can be selected from three different clocks.
- Two operating modes are available: reload timer mode (continuous operation) and one-shot mode (one-time operation).

Table 8.1-1 "Interval time range" lists the available interval time and square wave output ranges.

Table 8.1-1 Interval time range

Internal count clock cycle	Interval time	Square wave output (Hz)
1 t_{inst}	1 t_{inst} to 2^8 t_{inst}	$1/(2\ t_{inst})$ to $1/(2^9\ t_{inst})$
4 t_{inst}	2^2 t_{inst} to 2^{10} t_{inst}	$1/(2^3\ t_{inst})$ to $1/(2^{11}\ t_{inst})$
32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}	$1/(2^6\ t_{inst})$ to $1/(2^{14}\ t_{inst})$

t_{inst}: Instruction cycle (divide-by-four main clock oscillation)

The following shows an example of the interval time.

For a 5 MHz main clock oscillation (F_{CH}), a PWC reload buffer register (RLBR) value of "DD_H (221)", and a count clock cycle of one instruction cycle, the interval time and square wave output frequency are calculated as follows:

Interval time = (1 x 4/F_{CH}) x (RLBR register value)
= (4/5 MHz) x 221
= 176.8 μs

RLBR register value of "00_H" is assumed as 256.

■ Pulse width measurement function

The pulse width measurement function can measure the "H" width, "L" width, and one-cycle width of pulses input from an external pin (PWC pin).

- The PWC can perform continuous pulse width measurement.
- The measurement speed (internal count clock) can be selected from three different speeds.
- The width of long input pulses can be measured using an interrupt processing routine.

Table 8.1-2 "Available pulse width measured by pulse width measurement function" lists the available pulse widths measured by the pulse width measurement function.

Table 8.1-2 Available pulse width measured by pulse width measurement function

Internal count clock cycle	Interval time
1 t_{inst}	1 t_{inst} to 2^8 t_{inst}
4 t_{inst}	2^2 t_{inst} to 2^{10} t_{inst}
32 t_{inst}	2^5 t_{inst} to 2^{13} t_{inst}

t_{inst}: Instruction cycle (divide-by-four main clock oscillation)

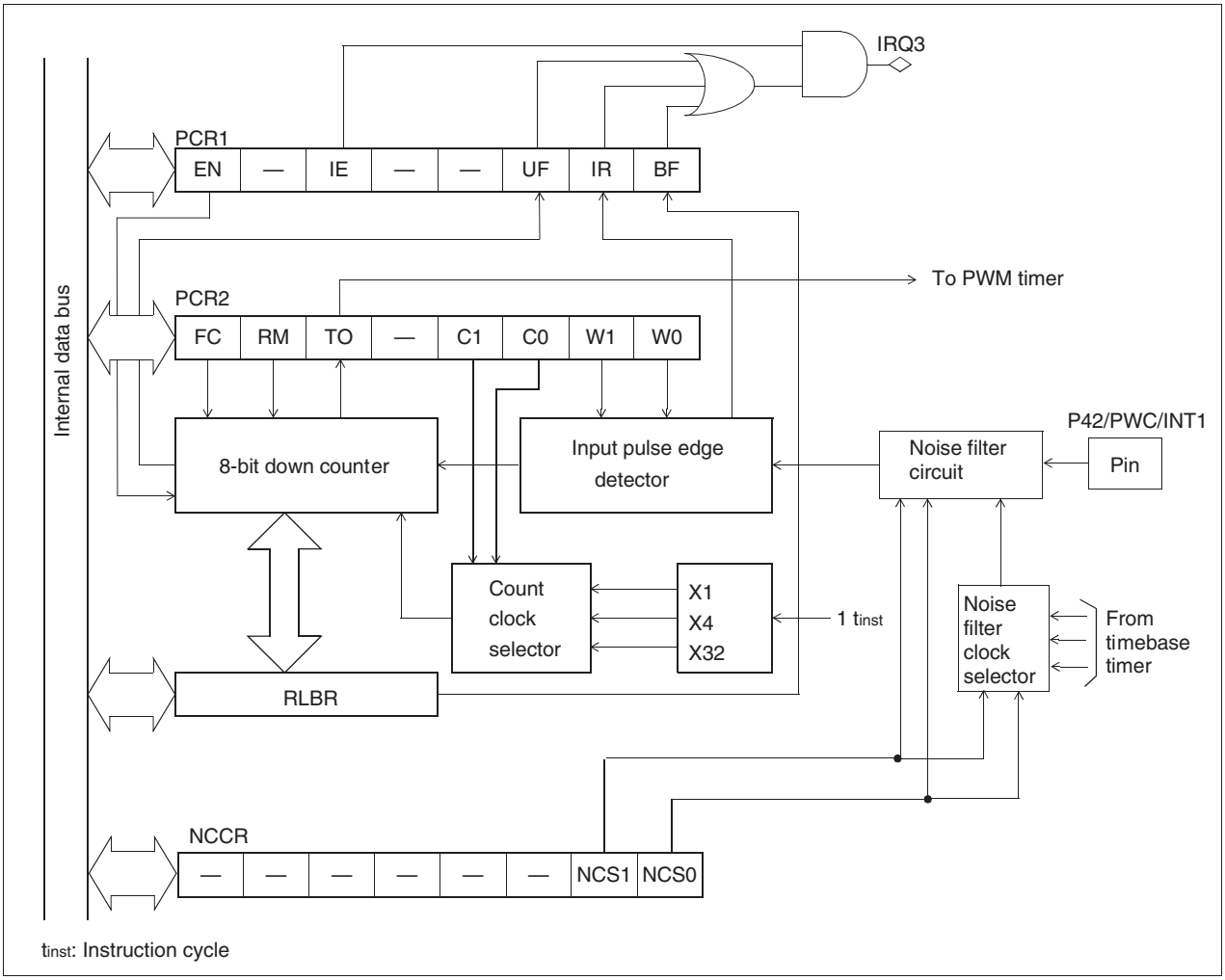
8.2 Block Diagram of Pulse Width Count Timer

The pulse width count timer consists of the following nine blocks:

- **Count clock selector**
- **8-bit down counter**
- **Input pulse edge detector**
- **Noise filter circuit**
- **Noise filter clock selector**
- **PWC reload buffer register (RLBR)**
- **PWC pulse width control register 1 (PCR1)**
- **PWC pulse width control register 2 (PCR2)**
- **Noise filter control register (NCCR)**

■ Block diagram of pulse width count timer

Figure 8.2-1 Block diagram of pulse width count timer



- Count clock selector

 Selects a count clock for the 8-bit down counter from the three available internal count clocks.

- 8-bit down counter

 The 8-bit down counter starts to count from the value set in the PWC reload buffer register (RLBR) when operating as an interval timer, and from FF_H when performing pulse width measurement. When an underflow (01_H --> 00_H) occurs, the counter inverts the timer output bit (PCR2: TO).

- Input pulse edge detector

 Operates when the pulse width measurement function is selected, and starts or stops the 8-bit down counter when an edge input from the PWC pin matches the edge specified by the PWC pulse width control register 2 (PCR2).

- Noise filter circuit

 The PWC input is sampled by the clock pulse selected by the sample clock selector. The sample input signal is integrated to clear the noise.

- Noise filter clock selector

 Selects a sampling clock for the noise filter circuit from three count clocks of timebase timer.

- RLBR register

 When operating in reload timer mode of the interval timer function, the RLBR register value is re-loaded to the counter and the count continues whenever a counter value underflow (01_H --> 00_H) occurs.

 When performing pulse width measurement, the value of the 8-bit down counter is transferred to the RLBR register when measurement completes.

- PCR1 and PCR2 register

 These registers are used to select the function, set operating conditions, enable or disable operation, control interrupts, and to check the PWC status.

- NCCR register

 This register is used to select sampling clock pulse for the noise filter circuit.

8.3 Structure of Pulse Width Count Timer

This section describes the pins, pin block diagram, registers, and interrupt source of the pulse width count timer.

■ Pulse width count timer pin

The pulse width count timer uses the P42/PWC/INT1 pin. This pin can function either as CMOS general-purpose I/O port (P42) or external interrupt (INT1), or as the measured pulse input (PWC).

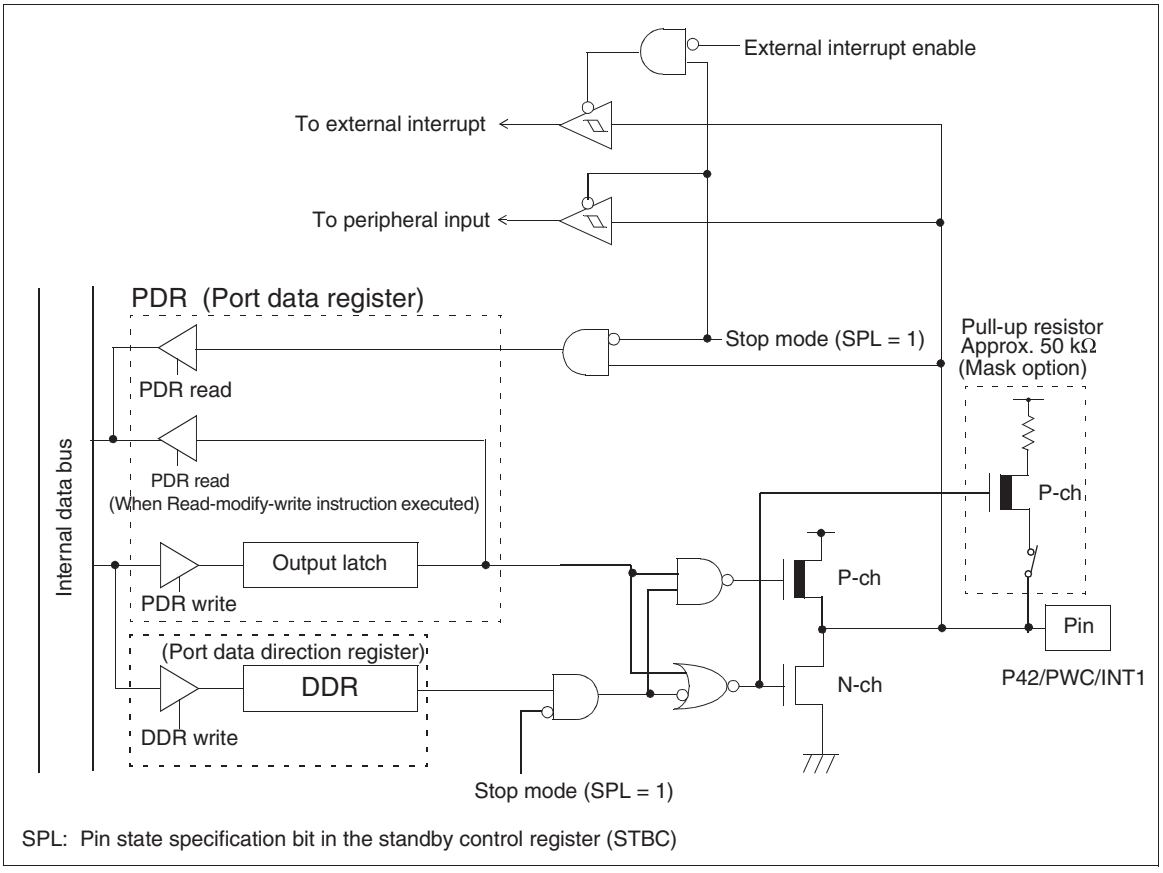
PWC:

The pulse width measurement function measures the pulse widths input to this pin.

Set the pin as an input port in the port data direction register (DDR4: bit 2 = "0") when using as the PWC pin for the pulse width measurement function.

■ Block diagram of pulse width count timer pin

Figure 8.3-1 Block diagram of pulse width count timer pin



SPL: Pin state specification bit in the standby control register (STBC)

■ Pulse width count timer registers

Figure 8.3-2 Pulse width count timer registers

PCR1 (PWC pulse width control register 1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0014H	EN	—	IE	—	—	UF	IR	BF	0-0--000B
	R/W		R/W			R/W	R/W	R	

PCR2 (PWC pulse width control register 2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0015H	FC	RM	TO	—	C1	C0	W1	W0	000-0000B
	R/W	R/W	R/W		R/W	R/W	R/W	R/W	

RLBR (PWC reload buffer register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value	
0016H									XXXXXXXXB	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		For the interval timer function
	R	R	R	R	R	R	R	R		For the pulse width measurement function

NCCR (Noise filter control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0017H	—	—	—	—	—	—	NCS1	NCS0	------00B
							R/W	R/W	

R/W: Readable and writable
R : Read-only
— : Unused
X : Indeterminate

■ Pulse width count timer interrupt source

IRQ3:

For both the interval timer and pulse width measurement function, the PWC generates an interrupt request if interrupt request output is enabled (PCR1: IE = "1") when the counter value underflows (01_H --> 00_H).

For the pulse width measurement function, the PWC generates an interrupt request for the pulse width measurement function if interrupt request output is enabled (PCR1: IE = "1") when pulse width measurement completes or a pulse width measurement value remains in the RLBR register.

8.3.1 PWC Pulse Width Control Register 1 (PCR1)

The PWC pulse width control register 1 (PCR1) is used to enable or disable functions, control interrupts and check the state of the pulse width count timer.

■ PWC pulse width control register 1 (PCR1)

Figure 8.3-3 PWC pulse width control register 1 (PCR1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0014H	EN	—	IE	—	—	UF	IR	BF	0-0--000B
	R/W		R/W			R/W	R/W	R	

BF	Buffer full flag bit
0	No pulse width measurement value.
1	Pulse width measurement value.

IR	Measurement completion interrupt request flag bit	
	Read	Write
0	Pulse width measurement has not completed.	Clears this bit.
1	Pulse width measurement has completed.	No effect. The bit does not change.

UF	Underflow (01H → 00H) interrupt request flag bit	
	Read	Write
0	No underflow (01H → 00H) on counter.	Clears this bit.
1	Underflow (01H → 00H) on counter.	No effect. The bit does not change.

The UF, IR, and BF bits are interrupt request flag bits.

IE	Interrupt request enable bit
0	Disables interrupt request output.
1	Enables interrupt request output.

EN	Counter operation enable bit	
	Timer function	Pulse width measure function
0	Count disables/stops.	Pulse width measure disables/stops.
1	Count enables/starts.	Pulse width measure enables/starts.

R/W : Readable and writable
R : Read-only
— : Unused
□ : Initial value

Table 8.3-1 PWC pulse width control register 1 (PCR1) bits

Bit		Function
Bit 7	EN: Counter operation enable bit	• For the interval timer function: Writing "1" to this bit starts the counter to count down from the PWC reload buffer register (RLBR) value. Writing "0" to this bit stops the counter operation. • For the pulse width measurement function: Writing "1" to this bit enables measurement. The counter starts to count down from "FF_H" on detection of the specified edge on the measurement pulse. Writing "0" to this bit stops the counter operation. **Note:** If operation is disabled (EN = "0") during measurement in pulse width measurement mode, the counter stops but the value is not transferred to the RLBR register. Restarting operation (EN = "1") sets the counter value to "FF_H" then enables operation.
Bit 6	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 5	IE: Interrupt request enable bit	• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and one or more of the interrupt request flag bits (UF, IR, and BF) are "1".
Bit 4 Bit 3	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 2	UF: Underflow (01_H --> 00_H) interrupt request flag bit	• This bit is set to "1" when the counter underflow (01_H --> 00_H) occurs. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value. **Notes:** • When the interval timer function is active, the PWC inverts the timer output bit (PCR2: TO) if the counter underflow (01_H --> 00_H) occurs. In reload timer mode, counting down continues from the RLBR register value. In one-shot timer mode, the counter operation automatically stops (EN = "0"). • If the counter underflow (01_H --> 00_H) occurs while measuring a long input pulse in the pulse width measurement function, this bit is set to "1" and counter operation continues.
Bit 1	IR: Measurement completion interrupt request flag bit	• For the pulse width measurement function: This bit is set to "1" when the pulse width measurement is completed. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value. • For the interval timer function: The bit has no meaning.
Bit 0	BF: Buffer full flag bit	• For the pulse width measurement function: This bit is an interrupt request flag and is set to "1" when a measurement value is present in the RLBR register. An interrupt request is output when both this bit and the interrupt request enable bit (IE) are "1". This bit is set to "1" when pulse width measurement completes and cleared to "0" when the measurement value is read from the RLBR register. This bit is read-only. The write value has no meaning and has no effect on the operation. • For the interval timer function: This bit has no meaning.

8.3.2 PWC Pulse Width Control Register 2 (PCR2)

The PWC pulse width control register 2 (PCR2) is used to select the operating mode (pulse width measurement or interval timer operation, etc.), select the count clock, set the measured pulse (measurement edges), and check the timer output state of the pulse width count timer.

■ PWC pulse width control register 2 (PCR2)

Figure 8.3-4 PWC pulse width control register 2 (PCR2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0015H	FC	RM	TO	—	C1	C0	W1	W0	000-0000B
	R/W	R/W	R/W		R/W	R/W	R/W	R/W	

W1	W0	Measured pulse selection bits Only applies to the pulse width measurement function (FC = "1").
0	0	High level
0	1	Low level
1	0	Rising edge to rising edge
1	1	Falling edge to falling edge

C1	C0	Count clock selection bits
0	0	1 t_{inst}
0	1	4 t_{inst}
1	0	32 t_{inst}
1	1	Do not use this setting.

t_{inst}: Instruction cycle

TO	Timer output bit	
	Read	Write
0	Reads the current output value.	Can be used to set the output value when the counter is stopped.
1		

RM	Reload mode selection bit Only applies to the interval timer function (FC = "0").
0	Reload timer mode
1	One-shot timer mode

FC	Operating mode selection bit
0	Operates as an interval timer function.
1	Operates as a pulse width measurement function.

R/W : Readable and writable
— : Unused
□ : Initial value

Table 8.3-2 PWC pulse width control register 2 (PCR2) bits

Bit		Function
Bit 7	FC: Operating mode selection bit	• This bit switches between the interval timer function (FC = "0") and pulse width measurement function (FC = "1"). **Note:** When using the pulse width measurement function (FC = "1"), set the P42/PWC/INT1 pin as an input port.
Bit 6	RM: Reload mode selection bit	• For the interval timer function: This bit selects reload timer mode (RM = "0") or one-shot timer mode (RM = "1"). • For the pulse width measurement function: This bit has no meaning.
Bit 5	TO: Timer output bit	• The value of this bit is inverted each time a counter value underflow (01_H --> 00_H) occurs. • By counting the number of times this bit is inverted (number of underflow (01_H --> 00_H) occurs), pulse widths longer than 2^8 × the cycle of the selected count clock can be measured.
Bit 4	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 3 Bit 2	C1, C0: Count clock selection bits	• These bits select the count clock for the interval timer function and pulse width measurement function. • Three internal count clocks can be selected. **Note:** Do not set "11_B" to C1 and C0 bits.
Bit 1 Bit 0	W1, W0: Measured pulse selection bits	• For the pulse width measurement function: These bits select which pulse edges to use as the start and end conditions for pulse measurement. Four types of pulse width or cycle can be selected. • For the interval timer function: These bits have no meaning.

Note:
Do not modify the PCR2 register while the counter is operating (PCR1: EN = "1").

8.3.3 PWC Reload Buffer Register (RLBR)

The PWC reload buffer register (RLBR) functions as a reload register for the interval timer function and as a measurement value storage register for the pulse width measurement function.

■ PWC reload buffer register (RLBR)

Figure 8.3-5 "PWC reload buffer register (RLBR)" shows the bit structure of the PWC reload buffer register.

Figure 8.3-5 PWC reload buffer register (RLBR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0016H									XXXXXXXXB
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	For the interval timer function
	R	R	R	R	R	R	R	R	For the pulse width measurement function

R/W: Readable and writable
R : Read-only
X : Indeterminate

● For interval timer function

The register functions as a reload register, specifying the interval time.

The counter starts to count down from the set value written in this register when counter operation is enabled (PCR1: EN = "1").

In reload timer mode, the RLBR register value is reloaded to the counter and the counter continues counting down when a counter value underflows (01_H --> 00_H). If a value is written to the RLBR register during counter operation, the new value applies from the next time the counter is reloaded due to an underflow (01_H --> 00_H).

Reference:

The setting value of the RLBR register for the interval timer function is calculated as follows:

RLBR register value = interval time/(count clock cycle x instruction cycle)

● For pulse width measurement function

The register is used to store the pulse width measurement value.

The counter value is transferred to this register when pulse width measurement completes on detection of the edge specified for measurement completion.

At this time, the buffer full flag bit (PCR1: BF) and the measurement completion interrupt request flag bit (PCR1: IR) are set to "1". Reading this register clears the BF bit to "0".

The register is read-only if the pulse width measurement function is selected.

Reference:

The pulse width for the pulse width measurement function is calculated based on the RLBR register value as follows:

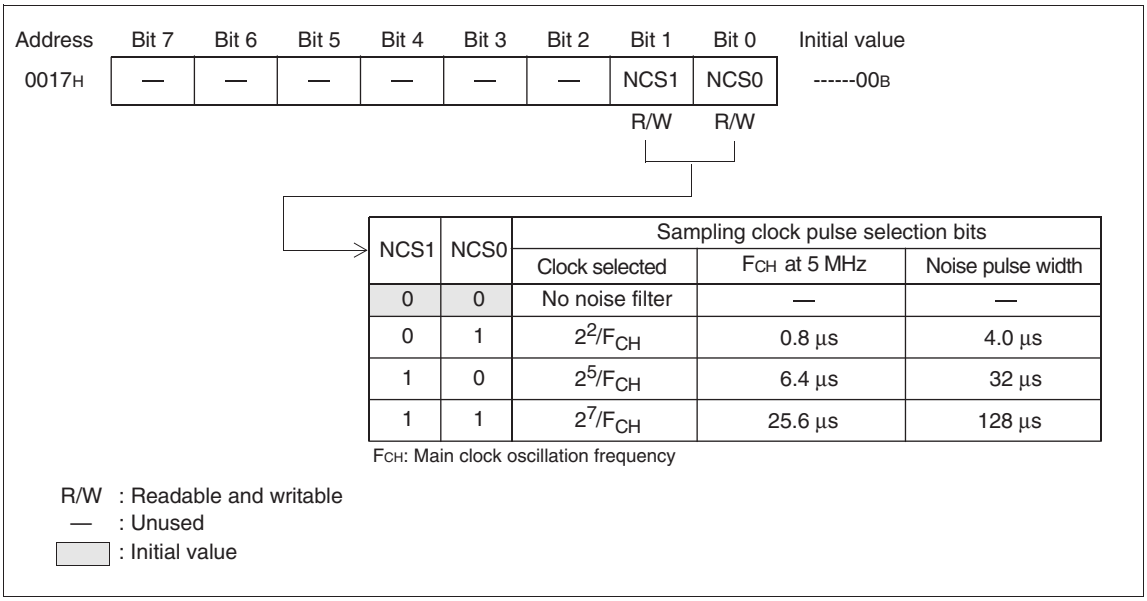
Pulse width = (256 - RLBR register value) x count clock cycle x instruction cycle

8.3.4 PWC Noise Filter Control Register (NCCR)

The PWC noise filter control register is used to select the sampling clock for the noise filter circuit. There are three type of selectable sampling clock from the timebase timer.

■ PWC noise filter control register (NCCR)

Figure 8.3-6 PWC noise filter control register (NCCR)



NCS1	NCS0	Sampling clock pulse selection bits		
		Clock selected	F_{CH} at 5 MHz	Noise pulse width
0	0	No noise filter	—	—
0	1	$2^2/F_{CH}$	0.8 µs	4.0 µs
1	0	$2^5/F_{CH}$	6.4 µs	32 µs
1	1	$2^7/F_{CH}$	25.6 µs	128 µs

F_{CH}: Main clock oscillation frequency

Table 8.3-3 PWC noise filter control register (NCCR) bits

Bit		Function
Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 1 Bit 0	NCS1, NCS0: Sampling clock pulse selection bits	• For the pulse width measurement function: These bits select sampling clock pulse for the noise filter circuit. There are three type of selectable sampling clock pulse from timebase timer. • For the interval timer function: These bits have no meaning.

8.4 Pulse Width Count Timer Interrupts

The pulse width count timer has the following two interrupts:

- **Counter value underflow (01_H --> 00_H) for the interval timer function**
- **Measurement completion and buffer full for the pulse width measurement function**

■ Interrupt for the interval timer function

The counter counts down from the set value on the selected internal count clock. When an underflow occurs, the underflow (01_H --> 00_H) interrupt request flag bit (PCR1: UF) is set to "1". At this time, an interrupt request (IRQ3) to the CPU is generated if the interrupt request enable bit is enabled (PCR1: IE = "1"). Write "0" to the UF bit in the interrupt processing routine to clear the interrupt request.

References:

- The UF bit is not set if the counter is stopped (PCR1: EN = "0") at the same time as the counter value underflows (01_H --> 00_H).
- An interrupt request is generated immediately if the UF bit is "1" when the IE bit is changed from disabled to enabled ("0" --> "1").

■ Interrupt for pulse width measurement function

When the specified measurement completion edge is detected, the measurement completion interrupt request flag bit (PCR1: IR) and the buffer full flag bit (PCR1: BF) are set to "1". Also, when a counter underflow (01_H --> 00_H) occurs due to measurement of a long pulse, the UF bit is set to "1". At this time, an interrupt request (IRQ3) to the CPU is generated if the interrupt request enable bit is enabled (PCR1: IE = "1"). Write "0" to the IR and UF bit in the interrupt processing routine to clear the interrupt request. Also read the PWC reload buffer register (RLBR) to clear the BF bit to "0".

References:

- The IR and BF bit are not set if the counter is stopped (PCR1: EN = "0") at the same time as the specified measurement completion edge is detected.
- An interrupt request is generated immediately if the IR, BF, or UF bit is "1" when the IE bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for pulse width count timer interrupt

Table 8.4-1 Register and vector table for pulse width count timer interrupt

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ3	ILR1 ($007C_H$)	L31 (Bit 7)	L30 (Bit 6)	$FFF4_H$	$FFF5_H$

See Section 3.4-2 "Interrupt Processing" for details on the operation of interrupt.

8.5 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the pulse width count timer.

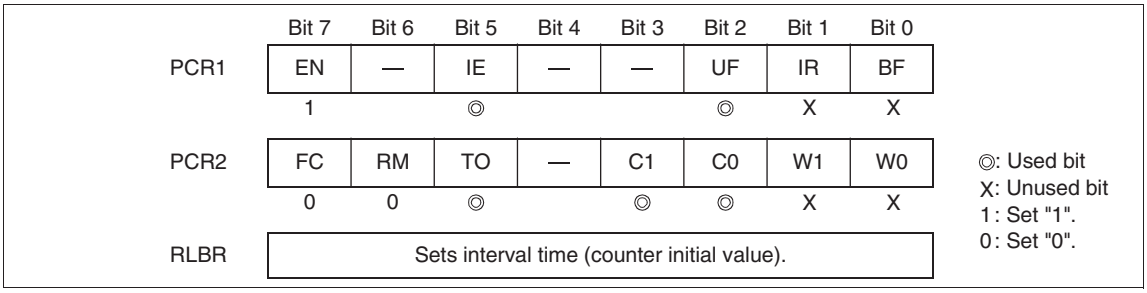
■ Operation of interval timer function

The interval timer function can operate as a continuous timer (reload timer mode), or as a timer that operates for one timer-cycle and then stops (one-shot mode).

● Reload timer mode

Figure 8.5-1 "Interval timer function (reload timer mode) settings" shows the settings required to operate in reload timer mode.

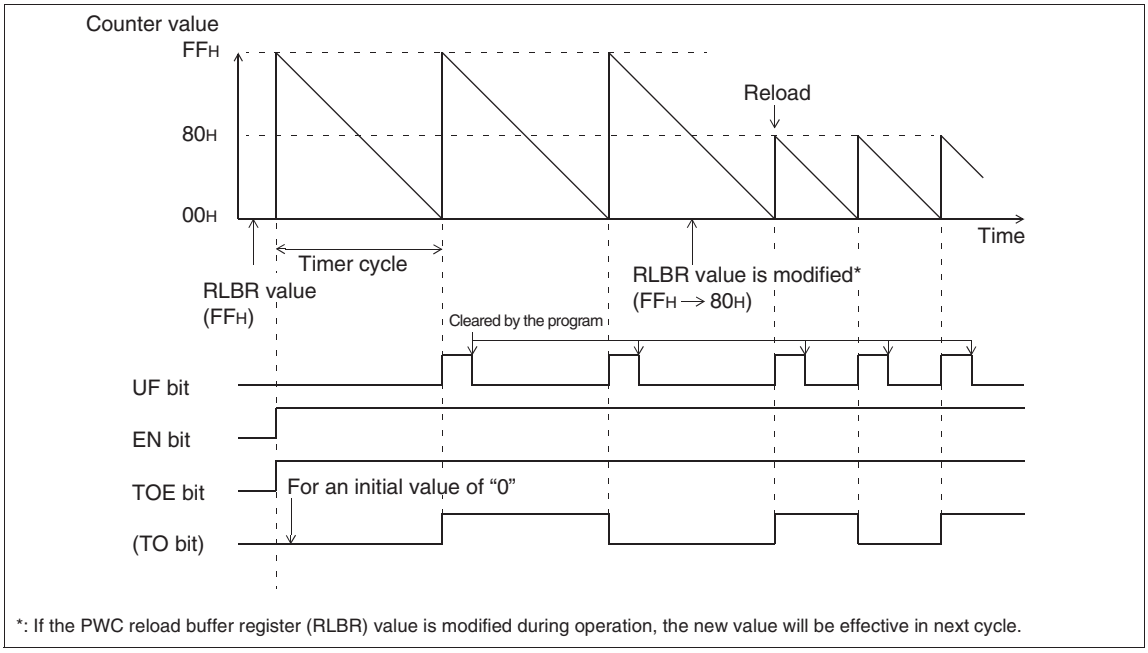
Figure 8.5-1 Interval timer function (reload timer mode) settings



On activation, the RLBR register value is loaded to the counter and the counter starts to count down on the rising edge of the selected count clock. When the counter value underflows (01_H --> 00_H), the PWC inverts the timer output bit (PCR2: TO) value, reloads the RLBR register value to the counter, and sets the underflow (01_H --> 00_H) interrupt request flag bit (PCR1: UF = "1") on the next rising edge of the count clock.

Figure 8.5-2 "Operation in reload timer mode" shows the operation in reload timer mode.

Figure 8.5-2 Operation in reload timer mode



Reference:

Setting the RLBR register value to "01_H" causes the TO bit to be inverted after each count clock cycle.

● One-shot timer mode

Figure 8.5-3 "Interval timer function (one-shot timer mode) settings" shows the settings required to operate in one-shot timer mode.

Figure 8.5-3 Interval timer function (one-shot timer mode) settings

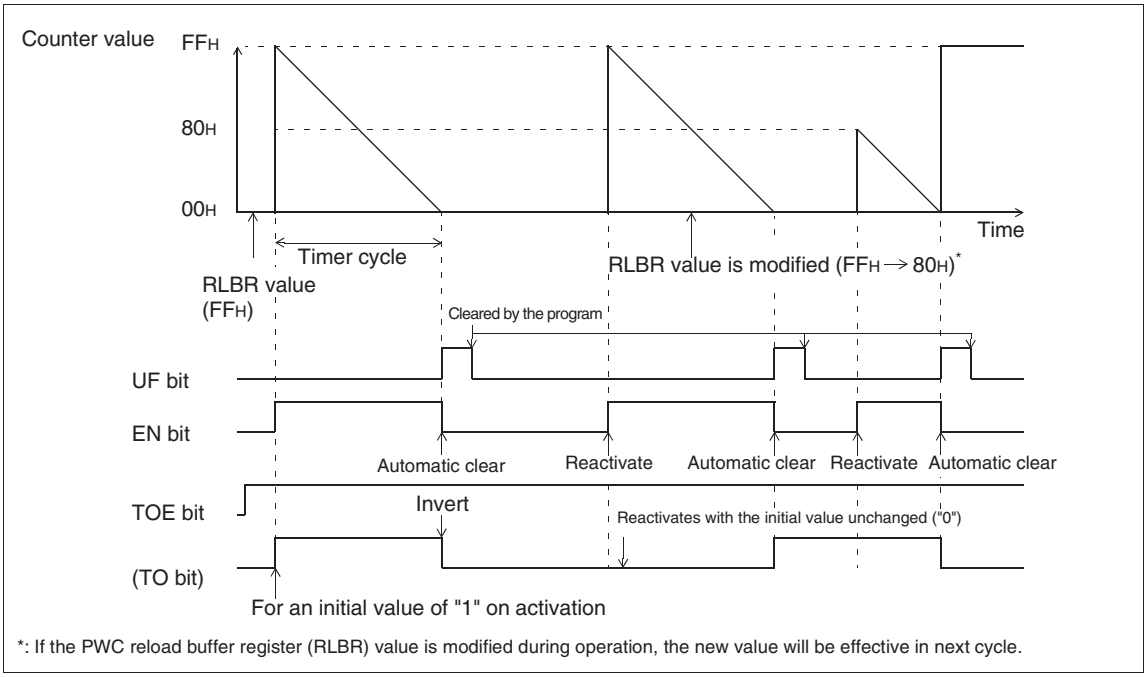
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCR1	EN	—	IE	—	—	UF	IR	BF
	1		◎			◎	X	X
PCR2	FC	RM	TO	—	C1	C0	W1	W0
	0	1	◎		◎	◎	X	X
RLBR	Sets interval time (counter initial value).							

◎: Used bit
X: Unused bit
1: Set "1".
0: Set "0".

On activation, the RLBR register value is loaded to the counter and the counter starts to count down on the rising edge of the selected count clock. When the counter value underflows (01_H --> 00_H), the counter inverts the timer output bit (PCR2: TO) value, automatically clears the counter operation enable bit (PCR1: EN = "0") to stop counter operation, and sets the underflow (01_H --> 00_H) interrupt request flag bit (PCR1: UF = "1") on the next rising edge of the count clock.

Figure 8.5-4 "Operation in one-shot timer mode" shows the operation in one-shot timer mode.

Figure 8.5-4 Operation in one-shot timer mode



*: If the PWC reload buffer register (RLBR) value is modified during operation, the new value will be effective in next cycle.

Note:

Do not modify PCR2 when the counter is operating (PCR1: EN = "1").

References:

- The UF bit is set to "1" if counter underflows (01_H --> 00_H), regardless of the value of the interrupt request enable bit (PCR1: IE).
- When the counter is stopped (PCR1: EN = "0") while the interval timer function is selected, the TO bit maintains the value it had immediately before the counter stopped.

8.6 Operation of Pulse Width Measurement Function

This section describes the operations of the pulse width measurement function of the pulse width count timer.

■ Operation of pulse width measurement function

Figure 8.6-1 "Pulse width measurement function settings" shows the settings required to operate as the pulse width measurement function.

Figure 8.6-1 Pulse width measurement function settings

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DDR4								
	X	X	X	X	X	0	X	X
PCR1	EN	—	IE	—	—	UF	IR	BF
	1		◎			△	◎	◎
PCR2	FC	RM	TO	—	C1	C0	W1	W0
	1	X	X		◎	◎	◎	◎
RLBR	Holds the pulse width measurement value.							

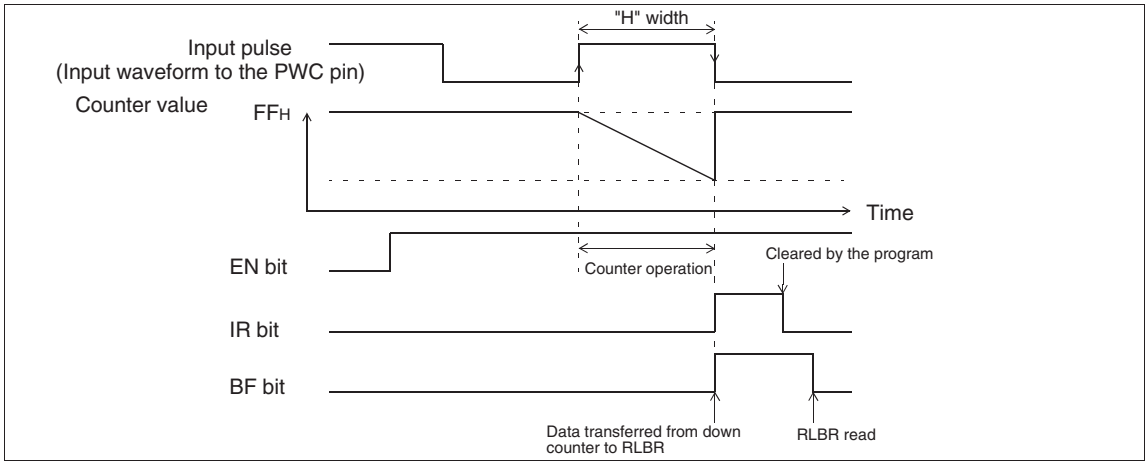
◎: Used bit
△: Used to measure long pulse widths
X: Unused bit
1: Set "1".
0: Set "0".

When counter operation is enabled, the counter starts to count down from "FF_H" when a measurement start edge is detected on the pulse input to the PWC pin. (For "H" level measurement, the counter starts measurement from the next rising edge if the input is already "H".)

On detection of the measurement completion edge, the current down counter value is transferred to the PWC reload buffer register (RLBR), the measurement completion interrupt request flag bit (PCR1: IR) and buffer full flag bit (PCR1: BF) are both set to "1", and counter operation is re-enabled. (The function supports continuous pulse width measurement and so can be used like an input capture.)

Figure 8.6-2 "Example of "H" width measurement using pulse width measurement function" shows the operation when the measured pulse selection bits (PCR2: W1, W0) are set to "00_B" ("H" width measurement).

Figure 8.6-2 Example of "H" width measurement using pulse width measurement function



Notes:

- If the previous RLBR register value has not been read during continuous pulse width measurement, the PWC leaves the BF bit set to "1" and maintains the previous measurement value. In this case, the new measurement value is lost.
- Do not modify the PCR2 register during pulse width measurement (PCR1: EN = "1").

■ Measuring long pulse widths

To measure pulse widths longer than 2^8 times the cycle of the selected count clock, it is necessary to count the number of counter underflows (01_H --> 00_H) by software in the interrupt processing routine. Counting by software requires a buffer in RAM (a software counter) to hold the number of counter underflows (01_H --> 00_H).

After initializing the software counter and enabling counter operation, the counter starts to count down from "FF_H" when a measurement start edge is detected on the pulse input to the PWC pin.

An interrupt request is generated on detection of the measurement completion edge or when the counter underflows (01_H --> 00_H). Check the measurement completion interrupt request flag bit (PCR1: IR) and underflow (01_H --> 00_H) interrupt request flag bit (PCR1: UF) in the interrupt processing routine. If the UF bit is "1", write "0" to the UF bit to clear the interrupt request and increment the software counter (the PWC counter continues to operate).

When the IR bit is "1", calculate the pulse width (including underflows (including underflows (01_H --> 00_H) from the values of the software counter and PWC reload buffer register (RLBR).

When the RLBR register value is "00_H", calculate as 256.

● Calculating the width of long pulses

Pulse width = [(256 - RLBR register value) + (number of counter underflows (01_H --> 00_H) x 256)] x one-cycle width of count clock

Calculate the pulse width before the next underflow (01_H --> 00_H) occurs. The correct measurement value may not be able to be calculated after the next underflow (01_H --> 00_H) occurs.

Figure 8.6-3 "Measuring long pulse width (falling edge to falling edge)" shows the operation when the measured pulse selection bits (PCR2: W1, W0) are set to "11_B" (falling edge to falling edge).

Figure 8.6-3 Measuring long pulse width (falling edge to falling edge)

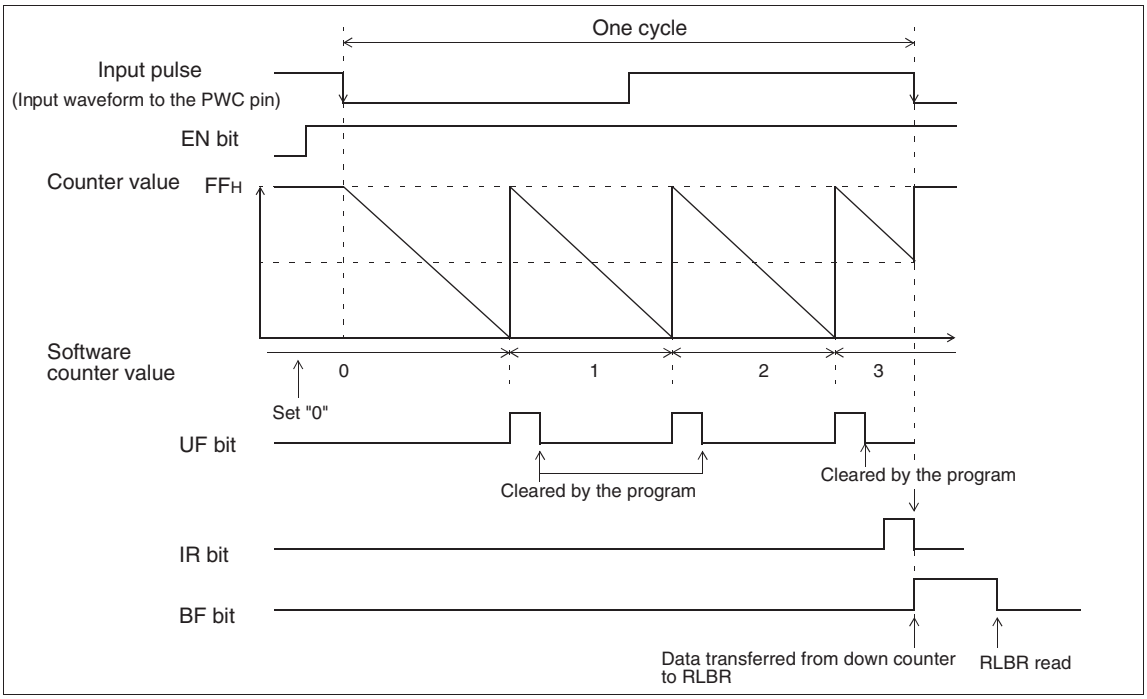
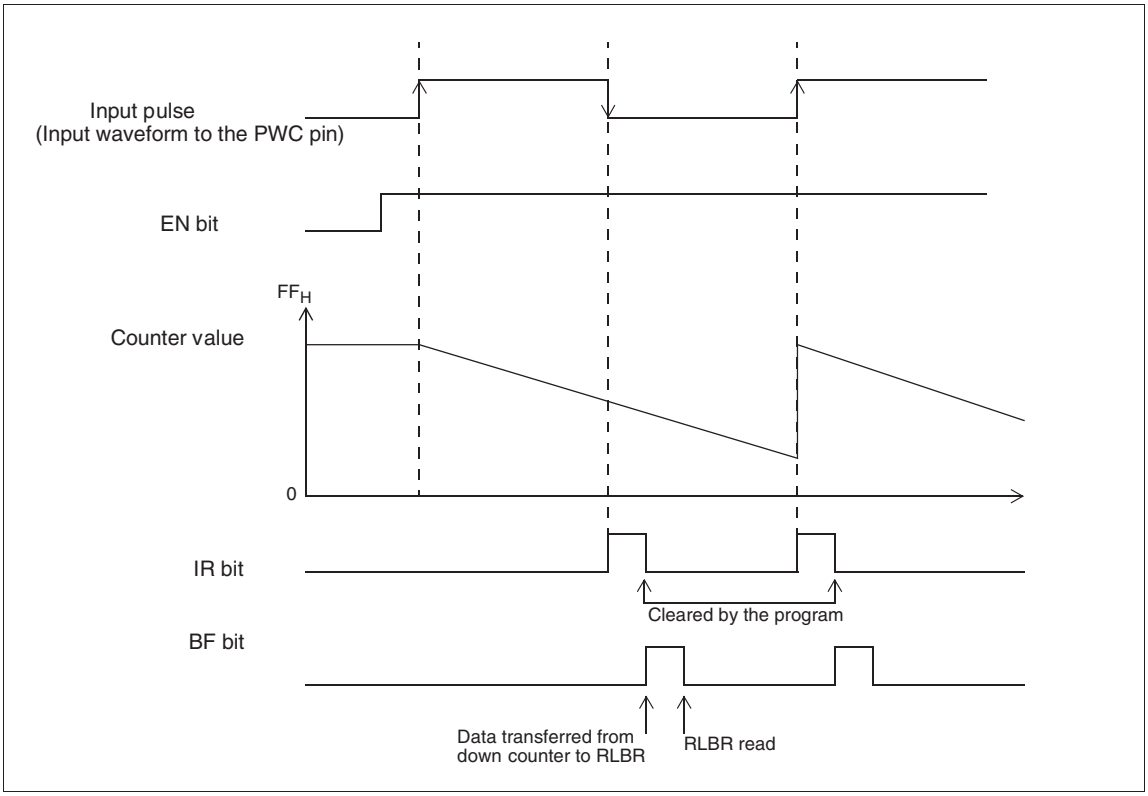


Figure 8.6-4 "Measuring long pulse width (rising edge to rising edge)" shows the operation when the measured pulse selection bits (PCR2: W1, W0) are set to "10_B" (rising edge to rising edge).

Figure 8.6-4 Measuring long pulse width (rising edge to rising edge)



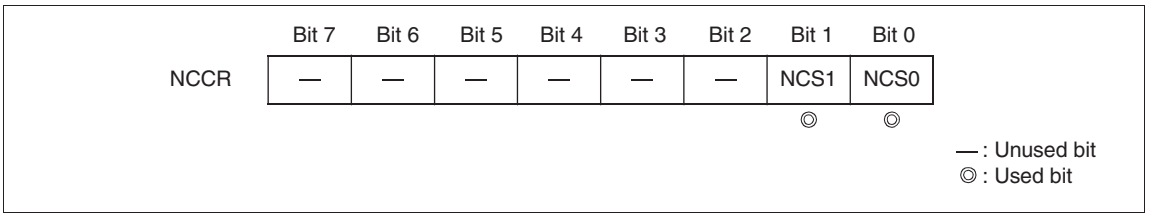
8.7 Operation of Noise Filter Circuit

This section describes the operations of noise filter circuit function when the pulse width measurement function is selected.

■ Operation of noise filter circuit function

Figure 8.7-1 "Noise filter circuit function settings" shows the settings required to operate as the noise filter circuit function.

Figure 8.7-1 Noise filter circuit function settings



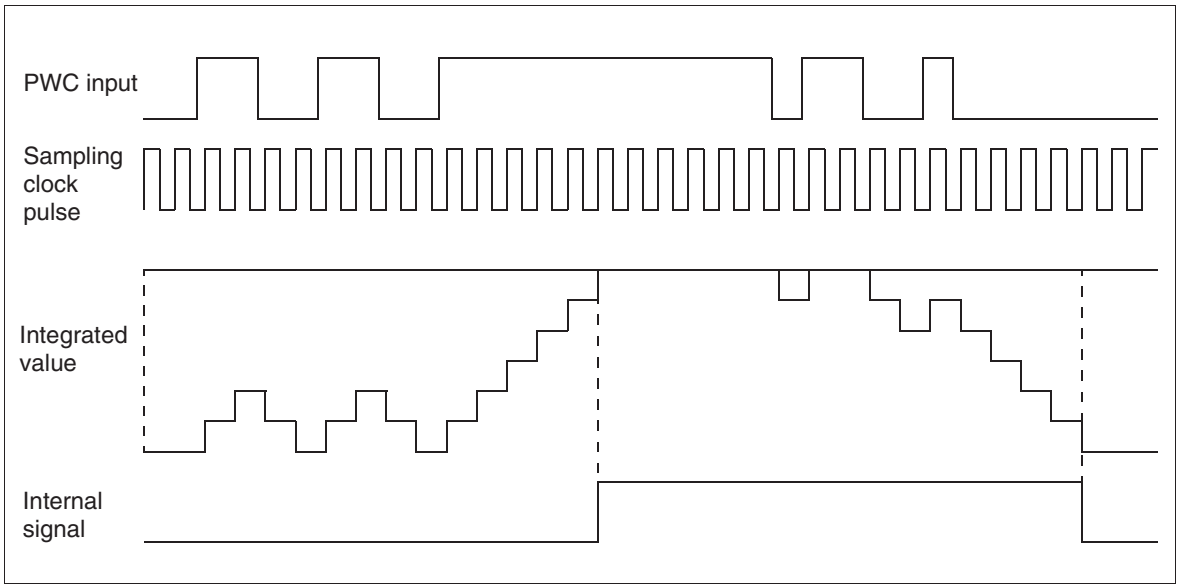
When pulse width measurement function is selected, the noise filter circuit can be used to clear the noise. By the selecting different value for sampling clock pulse selection bit (NCS1 and NCS0) of noise filter control register (NCCR), different kind of the noise can be filtered out. Integrating the sampled signal clears the noise. The maximum width of the cleared noise is as follows:

N_W = sampling clock cycle x 5

When noise clearing is prohibited, the PWC input is input directly to PWC counter/timer.

Figure 8.7-2 "Operation of noise filter circuit function" shows the operation of the noise filter circuit.

Figure 8.7-2 Operation of noise filter circuit function



8.8 States in Each Mode during Pulse Width Count Timer Operation

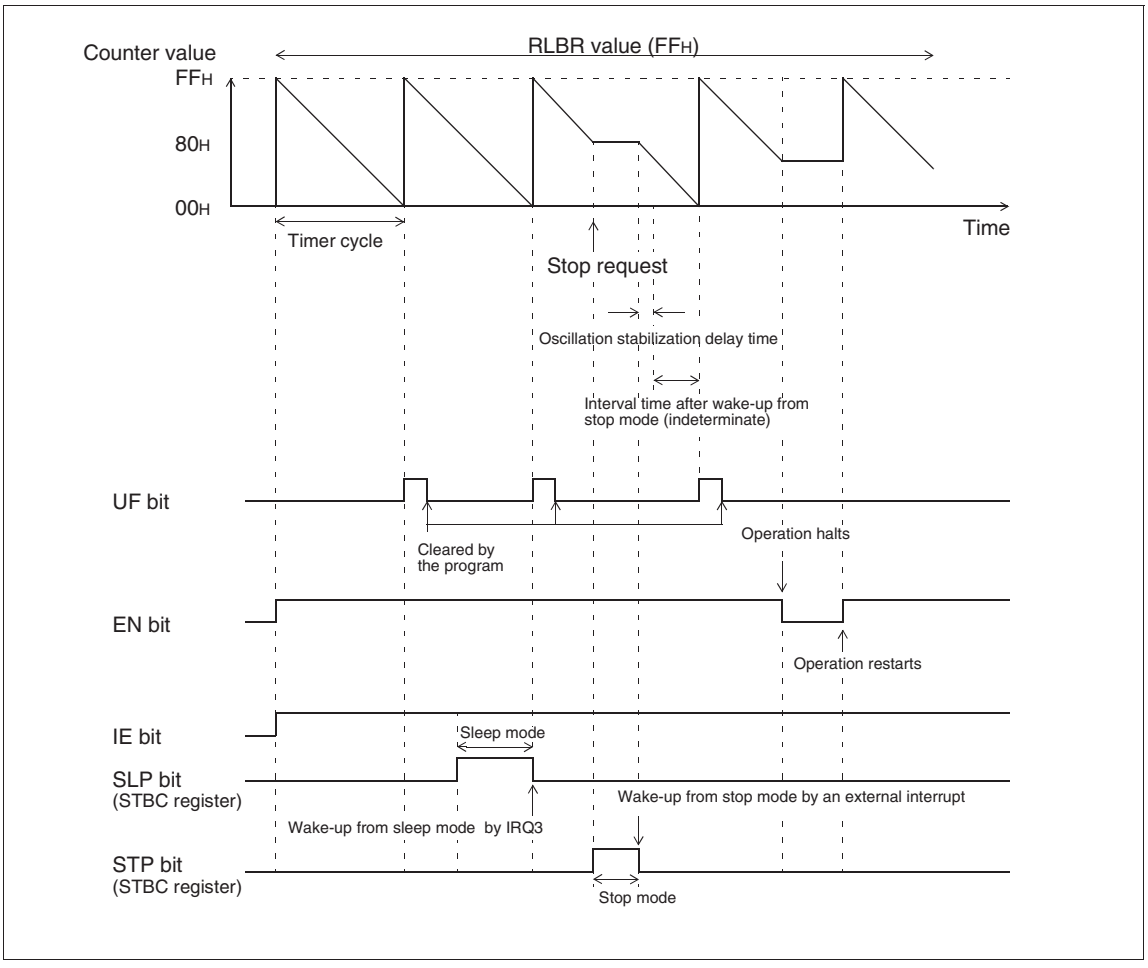
This section describes the operation of the pulse width count timer when the device goes to sleep or stop mode, or an operation halt request occurs during operation.

■ Operation during standby mode or operation halt

Figure 8.8-1 "Counter operation during standby mode or operation halt" shows the counter value state when the device goes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or pulse width measurement function.

The counter halts and maintains its current value when the device goes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. Therefore, the first interval time or pulse width measurement is not correct value. Always initialize the pulse width count timer after wake-up from stop mode.

Figure 8.8-1 Counter operation during standby mode or operation halt



8.9 Notes on Using Pulse Width Count Timer

This section lists points to note when using the pulse width count timer.

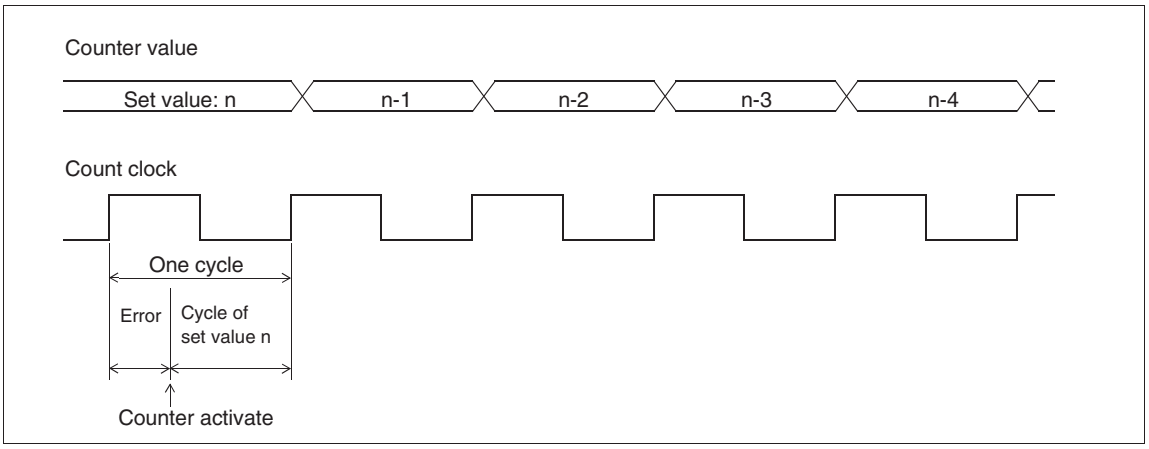
■ Notes on using pulse width count timer

● Error

When using the interval timer function, activating the counter by program is not synchronized with the start of counting-down using the selected internal count clock. Therefore, the time from activating the counter until an underflow occurs may be shorter than the theoretical time by a maximum of one cycle of the count clock.

Figure 8.9-1 "Error on starting counter operation" shows the error that occurs on starting counter operation.

Figure 8.9-1 Error on starting counter operation



● Notes on setting by program

- Do not modify the contents of the PWC pulse width control register 2 (PCR2) when the interval timer function or pulse width measurement function is operating (PCR1: EN = "1").
- Stop the counter (EN = "0"), disable interrupts (IE = "0"), and clear the interrupt request flag bits (UF, IR, BF = "000_B") in the PCR1 register before switching between the interval timer function and pulse width measurement function (PCR2: FC).
- Interrupt processing cannot return if the interrupt request flag bit (PCR1: UF, IR, or BF) is "1" and the interrupt request enable bit is enabled (PCR1: IE = "1"). Always clear the interrupt request flag bit.
- If a previous measurement value has not been read when performing continuous pulse width measurement for pulse width measurement function, new measurement values are not transferred to the PWC reload buffer register (RLBR). The RLBR maintains the previous value. Always read the measurement value before the next underflow (01_H --> 00_H) when measuring long pulse widths.
- The interrupt request flag bit (PCR1: UF, IR, or BF) is not set if the counter is disabled (PCR1: EN = "0") at the same time as an interrupt source is generated.

8.10 Program Example for Timer Function of Pulse Width Count Timer

This section gives two program examples for the timer function of the pulse width count timer.

■ Program example 1 for interval timer function (reload timer mode)

● Processing description

- Generates repeated interval timer interrupts at 3 ms intervals (reload timer mode).
- The TO bit will be inverted after each interval time cycle. The initial value of TO bit is "0" level.
- The following shows the RLBR register value that results in an interval time of approximately 3 ms for a 5 MHz main clock oscillation frequency. The count clock is 32 t_{inst} (t_{inst}: Instruction cycle).

 RLBR register value = 3 ms/(32 x 4/5 MHz) = 117.2 (075_H)

● Coding example

```
PCR1    EQU    0014H           ; Address of the PWC pulse width control register 1
PCR2    EQU    0015H           ; Address of the PWC pulse width control register 2
RLBR    EQU    0016H           ; Address of the PWC reload buffer register

EN      EQU    PCR1:7          ; Define the counter operation enable bit.
IE      EQU    PCR1:5          ; Define the interrupt request enable bit.
UF      EQU    PCR1:2          ; Define the underflow (01H → 00H) interrupt
                                 request flag bit.
BF      EQU    PCR1:0          ; Define the buffer full flag.

ILR1    EQU    007CH           ; Address of the interrupt level setting register 1

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF4H
IRQ3    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   EN              ; Stop counter operation.
        CLRB   IE              ; Disable interrupt request output.
        CLRB   BF              ; Clear buffer full flag (PCR1: bit 0).
        MOV    ILR1,#10111111B ; Set interrupt level (level 2).
        MOV    RLBR,#075H      ; Counter reload value (interval time)
        MOV    PCR2,#00001000B ; Select interval timer function, reload timer mode,
                                 initial output value of the TO bit, and 32 tinst.
        MOV    PCR1,#11100000B ; Start counter operation, enable interrupt
                                 request output, clear underflow (01H → 00H)
                                 interrupt request flag, clear measurement
                                 completion interrupt request flag (bit 1).
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   UF              ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

■ Program example 2 for interval timer function (one-shot timer mode)

● Processing description

- Generates a single 1.5 ms interval timer interrupt (one-shot timer mode).
- The TO bit is initialized to "1" and inverted after the interval time.
- The following shows the RLBR register value that results in an interval time of approximately 3 ms for a 5 MHz main clock oscillation frequency. The count clock is 32 t_{inst} (t_{inst}: Instruction cycle).

 RLBR register value = 3 ms/(32 x 4/5 MHz) = 117.2 (075_H)

● Coding example

```
PCR1    EQU    0014H           ; Address of the PWC pulse width control register 1
PCR2    EQU    0015H           ; Address of the PWC pulse width control register 2
RLBR    EQU    0016H           ; Address of the PWC reload buffer register

EN      EQU    PCR1:7          ; Define the counter operation enable bit.
IE      EQU    PCR1:5          ; Define the interrupt request enable bit.
UF      EQU    PCR1:2          ; Define the underflow (01H → 00H) interrupt
                                 request flag bit.
BF      EQU    PCR1:0          ; Define the buffer full flag

ILR1    EQU    007CH           ; Address of the interrupt level setting register 1

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF4H
IRQ3    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   EN              ; Stop counter operation.
        CLRB   IE              ; Disable interrupt request output.
        CLRB   BF              ; Clear buffer full flag (PCR1: bit 0).
        MOV    ILR1,#0111111B  ; Set interrupt level (level 1).
        MOV    RLBR,#075H      ; Counter reload value (interval time)
        MOV    PCR2,#01101000B ; Select interval timer function, one-shot timer
                                 mode, initial output value of the TO, and
                                 32 tinst.
        MOV    PCR1,#10100000B ; Start counter operation, enable interrupt request
                                 output, clear underflow (01H → 00H) interrupt
                                 request flag, clear measurement complete
                                 interrupt request flag (bit 1).
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   UF              ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 9

8-BIT SERIAL I/O

This chapter describes the functions and operation of the 8-bit serial I/O.

9.1 "Overview of 8-bit Serial I/O"

9.2 "Block Diagram of 8-bit Serial I/O"

9.3 "Structure of 8-bit Serial I/O"

9.4 "8-bit Serial I/O Interrupts"

9.5 "Operation of Serial Output"

9.6 "Operation of Serial Input"

9.7 "States in Each Mode during 8-bit Serial I/O Operation"

9.8 "Notes on Using 8-bit Serial I/O"

9.9 "Connection Example for 8-bit Serial I/O"

9.10 "Program Example for 8-bit Serial I/O"

9.1 Overview of 8-bit Serial I/O

The 8-bit serial I/O function is the serial transfer of 8-bit data, synchronized with the shift clock. The shift clock can be selected from one external and three internal clocks. The data shift direction can be selected as either LSB first or MSB first.

■ Serial I/O function

The 8-bit serial I/O function is the serial input and output of 8-bit data, synchronized with the shift clock.

- The serial I/O converts 8-bit parallel data to serial and outputs the serial data. Similarly, the serial I/O converts input serial data to parallel and stores the data.
- One shift clock can be selected from one external and three internal clocks.
- The serial I/O can control input and output of the shift clock and can output the internal shift clock.
- The data shift direction (transfer direction) can be selected as either LSB first or MSB first.

Table 9.1-1 Shift clock cycle and transfer speed

Shift clock	Clock (cycle)	Frequency (Hz)	Transfer speed (F_{CH} = 5 MHz)
Internal shift clock (output)	2 t_{inst}	1/(2 t_{inst})	625 kbps
	8 t_{inst}	1/(8 t_{inst})	156 kbps
	32 t_{inst}	1/(32 t_{inst})	39 kbps
External shift clock (input)	2 t_{inst} or more	1/(2 t_{inst}) or less	DC to 625 kbps

F_{CH}: main clock oscillation frequency

t_{inst}: Instruction cycle (1 instruction cycle = 4/F_{CH}).

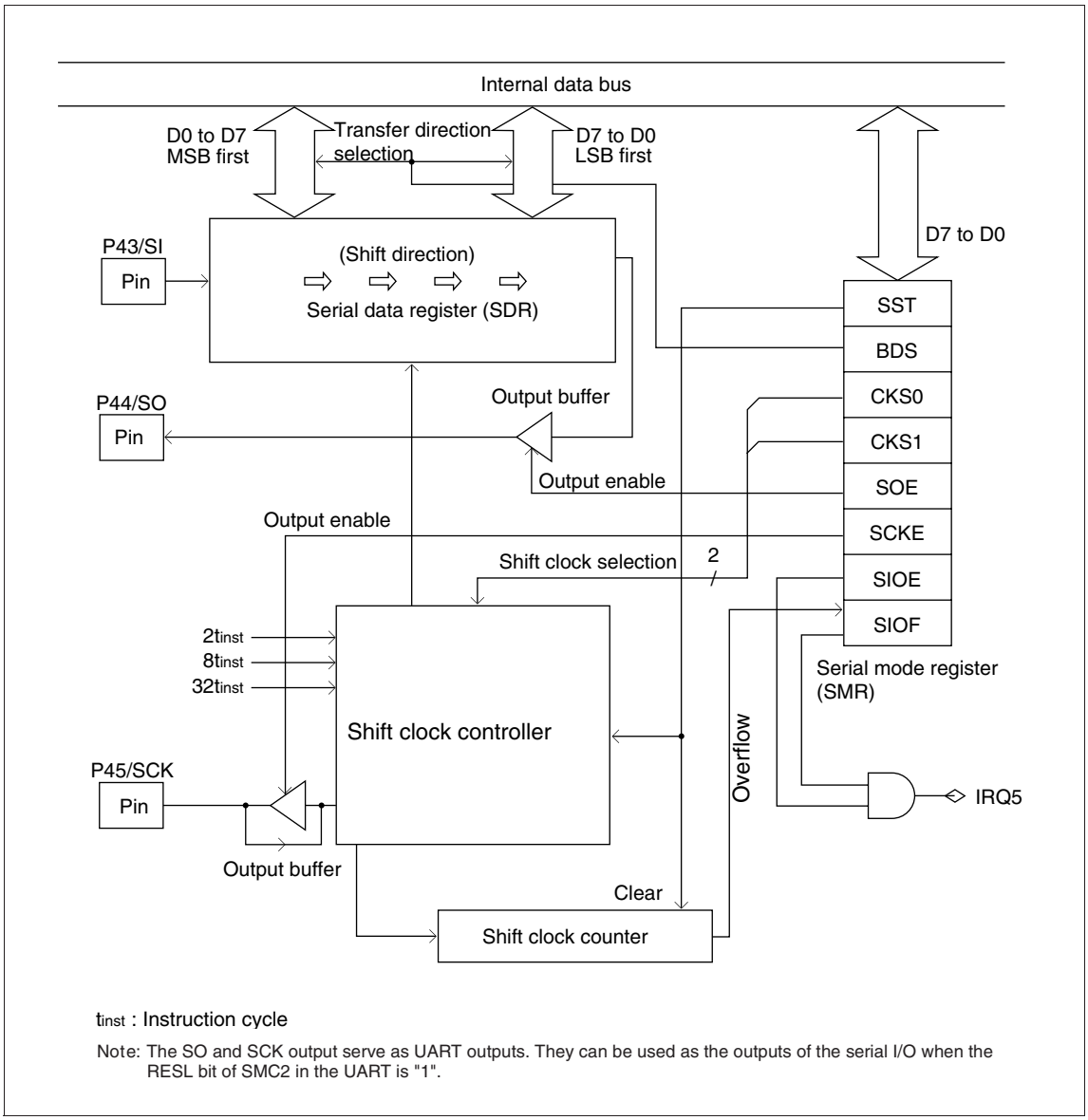
9.2 Block Diagram of 8-bit Serial I/O

Each channel of the 8-bit serial I/O consists of the following four blocks:

- **Shift clock controller**
- **Shift clock counter**
- **Serial data register (SDR)**
- **Serial mode register (SMR)**

■ Block diagram of 8-bit serial I/O

Figure 9.2-1 Block diagram of 8-bit serial I/O



t_{inst} : Instruction cycle

Note: The SO and SCK output serve as UART outputs. They can be used as the outputs of the serial I/O when the RESL bit of SMC2 in the UART is "1".

● Shift clock control circuit

Selects the shift clock from one external and three internal clocks.

If an internal shift clock is selected, the shift clock can be output to the SCK pin. If external shift clock is selected, the clock input from the SCK pin is used as the shift clock. The SDR register shifts in synchronous with the shift clock and the shifted-out value is output to the SO pin. Similarly, the serial input is obtained by shifting the SI pin input to the SDR register.

● Shift clock counter

The shift clock counter counts the number of SDR register shifts generated by the shift clock and overflows after eight shifts.

The overflow clears the serial I/O transfer start bit in the SMR register (SST = "0") and sets the interrupt request flag (SIOF = "1"). The shift clock counter stops counting when serial transfer halts (SST = "0"). The shift clock counter is cleared when serial transfer restarts (SST = "1").

● SDR register

The SDR register is used to store the transfer data. Data written to this register is converted to serial and output. Serial input is converted to parallel data and stored in this register.

● SMR register

The SMR register is used to enable or disable serial I/O operation, select the shift clock, set the transfer (shift) direction, control interrupts, and check the serial I/O status.

9.3 Structure of 8-bit Serial I/O

This section describes the pins, pin block diagram, registers, and interrupt source of 8-bit serial I/O.

■ 8-bit serial I/O pins

8-bit serial I/O uses the P43/SI, P44/SO, and P45/SCK. The pins are also used as UART I/O pins. To use the pins as serial I/O pins, set the UART/SIO selection bit RSEL of UART serial mode control register 2 (SMC2: RSEL = "1").

● P43/SI pin

The P43/SI pin can function either as a general-purpose I/O port (P43) or as the serial data input (hysteresis input) for 8-bit serial I/O or UART.

● P44/SO pin

The P44/SO pin can function either as a general-purpose I/O port (P44) or as the serial data output for 8-bit serial I/O or UART.

Enabling serial data output (SMR: SOE = "1" and UART/SIO selection bit SMC2: RSEL = "1") automatically sets the P44/SO pin as an output pin, regardless of the port data direction register (DDR4: bit 4) value, and sets the pin to function as the SO pin.

● P45/SCK pin

The P45/SCK pin can function either as a general-purpose I/O port (P45) or as the shift clock I/O for 8-bit serial I/O or UART.

Set P45/SCK pin as an input port in the data direction register (DDR4: bit 5 = "0") when using as SCK pin.

- When using as the shift clock input pin

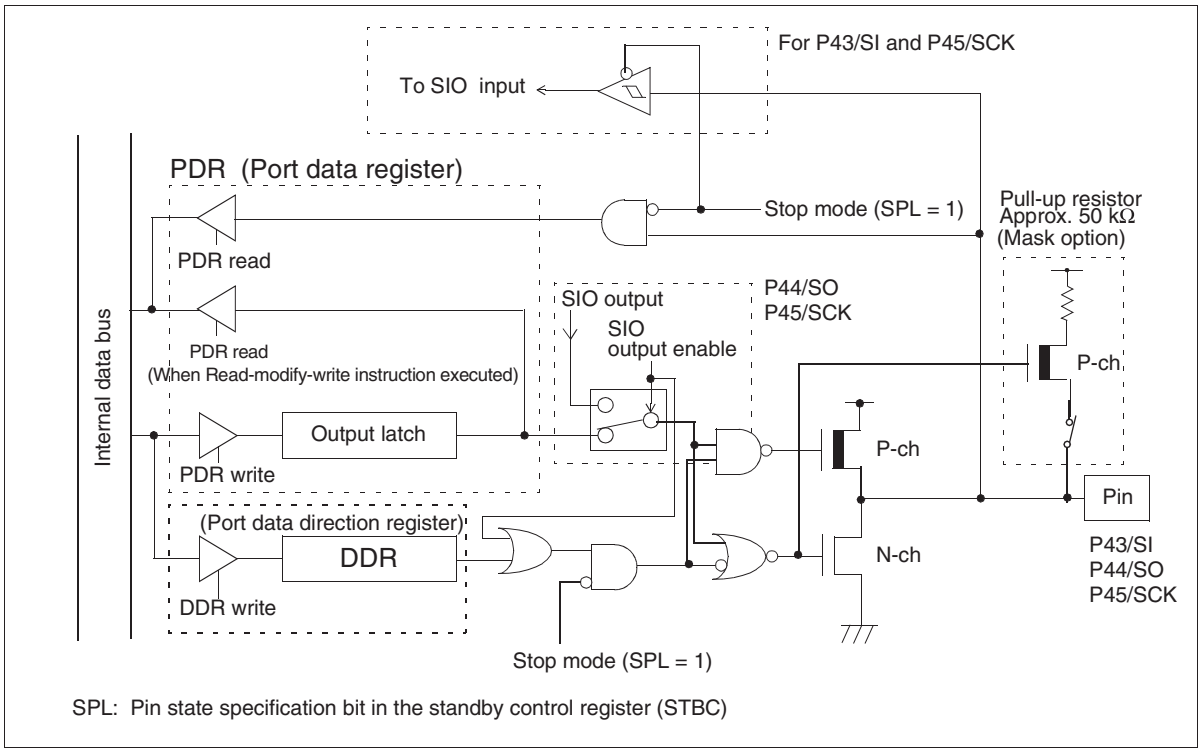
When using SCK as an input pin, set the pin as an input port in the port data direction register (DDR4: bit 5 = "0") and disable shift clock output (SMR: SCKE = "0"). In this case, always select external shift clock operation (SMR: CKS1, CKS0 = "11_B").

- When using as the shift clock output pin

Enabling shift clock output (SMR: SCKE = "1" and UART/SIO selection bit SMC2: RSEL = "1") automatically sets the P45/SCK pin as an output pin, regardless of the port data direction register (DDR4: bit 5) value, and sets the pin to function as the SCK output pin. In this case, always select internal shift clock operation (SMR: CKS1, CKS0 = other than "11_B").

■ Block diagram of 8-bit serial I/O pins

Figure 9.3-1 Block diagram of 8-bit serial I/O pin



SPL: Pin state specification bit in the standby control register (STBC)

Reference:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

■ 8-bit serial I/O registers

Figure 9.3-2 8-bit serial I/O registers

SMR (Serial mode register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001C$_H$	SIOF	SIOE	SCKE	SOE	CKS1	CKS0	BDS	SST	00000000$_B$
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SDR (Serial data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
001D$_H$									XXXXXXXX$_B$
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X : Indeterminate

■ 8-bit serial I/O interrupt source

IRQ5:

8-bit serial I/O generates an interrupt request (IRQ5) if interrupt request output is enabled (SMR: SIOE = "1") when the I/O function completes input or output of 8-bit serial data.

9.3.1 Serial Mode Register (SMR)

The serial mode register (SMR) is used to enable or disable operation, select the shift clock, set the transfer direction, control interrupts, and check the state of 8-bit serial I/O.

■ Serial mode register (SMR)

Figure 9.3-3 Serial mode register (SMR)

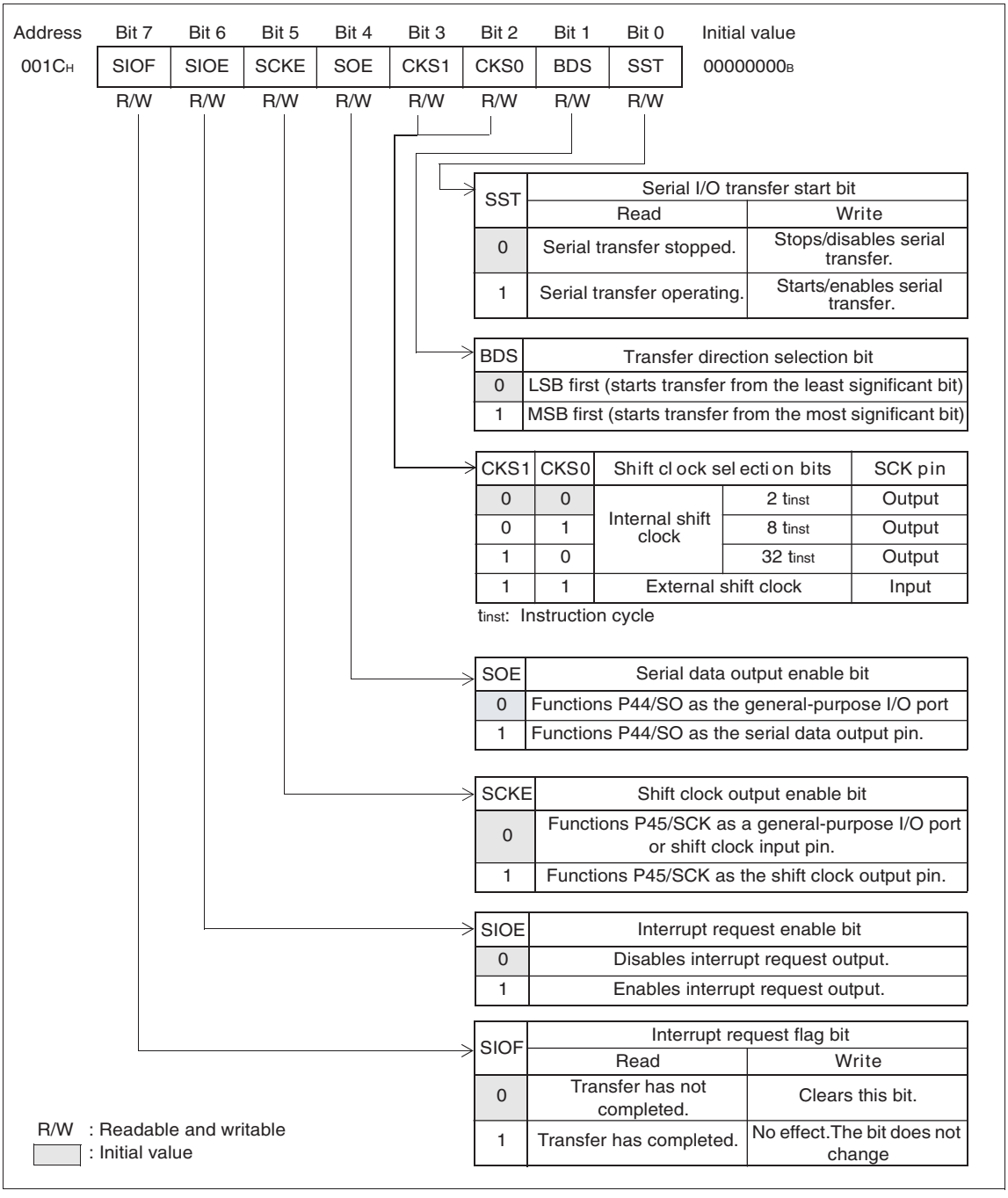


Table 9.3-1 Serial mode register (SMR) bits

Bit		Function
Bit 7	SIOF: Interrupt request flag bit	• This bit is set to "1" when the serial output operation has transmitted 8 serial data bits or the serial input operation has received 8 serial data bits. An interrupt request is generated when both this bit and the interrupt request enable bit (SIOE) are "1". • Writing "0" clears this bit. Writing "1" has no effect and does not change the bit value.
Bit 6	SIOE: Interrupt request enable bit	This bit enables or disables an interrupt request output to the CPU. An interrupt request is issued when both this bit and the interrupt request flag bit (SIOF) are "1".
Bit 5	SCKE: Shift clock output enable bit	• This bit controls shift clock input and output when UART/SIO selection bit (SMC2: RSEL) is set to "1". • The P45/SCK pin function as the shift clock input pin when this bit is set to "0" and as the shift clock output pin when this bit is set to "1". **Notes:** • Set the P45/SCK pin as an input port when using this pin as the shift clock input. Also, selects external shift clock operation in the shift clock selection bits (CKS1, CKS0 = "11_B"). • When using this pin as internal shift clock output (SCK = "1"), select internal shift clock operation (CKS1, CKS0 = other than "11_B"). **References:** • The pin functions as the SCK output pin when shift clock is enabled (SCKE = "1") regardless of the state of the general-purpose I/O port (P45). • Set to shift clock input operation (SCKE = "0") when using this pin as a general-purpose I/O port (P45).
Bit 4	SOE: Serial data output enable bit	• This bit controls serial data output when UART/SIO selection bit (SMC2: RSEL) is set to "1". • The P44/SO pin functions as a general-purpose I/O port (P44) when this bit is set to "0" and as the serial data output pin (SO) when this bit is set to "1". **Reference:** • The pin functions as the (SO) pin when serial data output is enabled (SOE = "1"), regardless of the state of the general-purpose I/O port (P44).
Bit 3 Bit 2	CKS1, CKS0: Shift clock selection bits	• These bits select the shift clock from one external and three internal shift clocks. • Setting these bits to other than "11_B" selects an internal shift clock. In this case, the shift clock is output from the SCK pin if the shift clock output enable bit (SCKE) is "1". • Setting these bits to "11_B" selects the external shift clock. This inputs the shift clock from the SCK pin if shift clock input is enabled (SCKE = "0" and DDR4: bit 5 = "0").
Bit 1	BDS: Transfer direction selection bit	• This bit selects whether serial data is transferred with the least significant bit first (LSB first, BDS = "0") or the most significant bit first (MSB first, BDS = "1"). **Note:** • As bits are set in the appropriate order when writing to or reading from the serial data register (SDR), modifying this bit does not apply to any data already set in the SDR register.

Table 9.3-1 Serial mode register (SMR) bits

Bit		Function
Bit 0	SST: Serial I/O transfer start bit	• This bit controls serial I/O transfer start and transfer enable. This bit can also be used to determine whether transfer has completed. • Writing "1" to this bit when an internal shift clock is selected (CKS1, CKS0 = other than "11_B") clears the shift clock counter and starts data transfer. • Writing "1" to this bit when an external shift clock is selected (CKS1, CKS0 = "11_B") enables data transfer, clears the shift clock counter, and sets serial I/O to delay for input of the external shift clock. • This bit is cleared to "0" and the SIOF bit is set to "1" when transfer completes. • Writing "0" to this bit while transfer is in progress (SST = "1") aborts the transfer. After halting a transfer, data must be set again to the SDR register for data output and transfer restarted (the shift clock counter cleared) for data input.

9.3.2 Serial Data Register (SDR)

**The serial data register (SDR) stores the transfer data for 8-bit serial I/O.
The register can function as the transmit data register for serial output operation or as the receive data register for serial input operation.**

■ Serial data register (SDR)

Figure 9.3-4 "Serial data register (SDR)" shows the bit structure of the serial data register.

Figure 9.3-4 Serial data register (SDR)

Register	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
SDR	001DH									XXXXXXXXB
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
X : Indeterminate

● Serial output operation

The register functions as the transmit data register. When serial I/O transfer starts (SMR: SST = "1"), the 8-bit serial I/O performs serial transfer of the data written in the register.

● Serial input operation

The register functions as the receive data register. When serial I/O transfer starts (SMR: SST = "1"), the received serial transfer data is stored in this register.

● During serial I/O transfer

Do not write data to the SDR register during serial I/O transfer operating. Also, the read value has no meaning.

9.4 8-bit Serial I/O Interrupts

The 8-bit serial I/O can generate interrupt requests after completion of the serial input and output of the 8-bit data.

■ Interrupt for serial output operation

The 8-bit serial I/O performs the serial input operation and serial output operation at the same time. When the serial transfer starts, the data in the serial data register (SDR) is input and output one bit at a time, synchronized with the cycle of the selected shift clock. The interrupt request flag bit (SMR: SIOF) is set to "1" on the rising edge of the shift clock of the eighth bit.

At this time, an interrupt request (IRQ5) to the CPU is generated if the interrupt request enable bit is enabled (SMR: SIOE = "1").

Write "0" to the SIOF bit in the interrupt processing routine to clear the interrupt request. The SIOF bit is set after completing 8-bit serial output, regardless of the SIOE bit value.

Reference:

The interrupt request flag bit is not set (SMR: SIOF = "1") if serial transfer is stopped (SMR: SST = "0") at the same time as serial data transfer completes for the serial I/O operation. An interrupt request is generated immediately if the SIOF bit is "1" when the SIOE bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for 8-bit serial I/O interrupts

Table 9.4-1 Register and vector table for 8-bit serial I/O interrupts

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ5	ILR2 ($007D_H$)	L51 (Bit 3)	L50 (Bit 2)	$FFF0_H$	$FFF1_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupt operation.

9.5 Operation of Serial Output

The 8-bit serial I/O can perform serial output of 8-bit data synchronized with a shift clock.

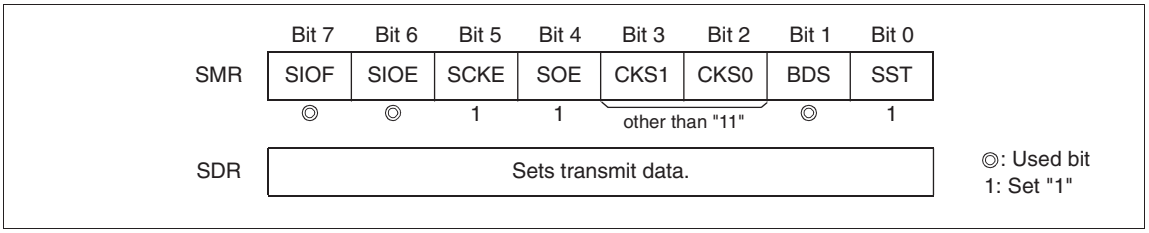
■ Serial output operation

Serial output can operate using an internal or external shift clock. When serial output operation is enabled, the contents of the SDR register are output to the serial data output pin (SO). Serial input is performed at the same time.

● Internal shift clock

Figure 9.5-1 "Serial output settings (when using internal shift clock)" shows the settings required to operate serial output using an internal shift clock.

Figure 9.5-1 Serial output settings (when using internal shift clock)

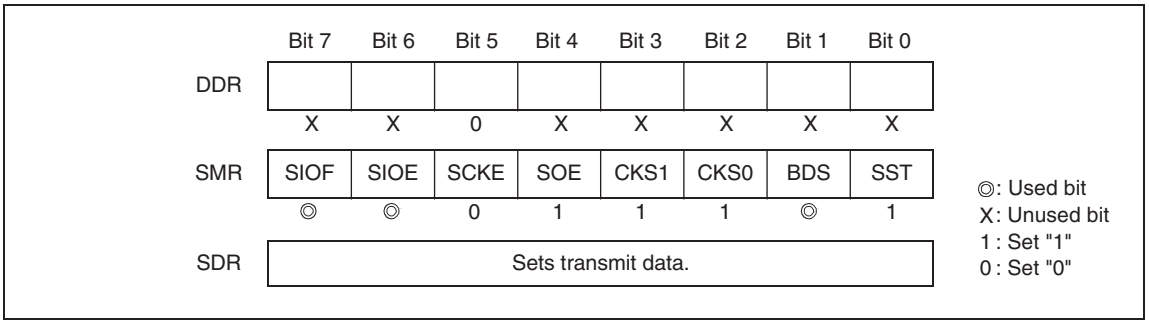


Activating serial output operation outputs the contents of the SDR register to the SO pin, synchronized with the falling edge of the selected internal shift clock. At this time, the device being communicated with (a serial input) must be waiting for input of the external shift clock.

● External shift clock

Figure 9.5-2 "Serial output settings (when using external shift clock)" shows the settings required to operate serial output using an external shift clock.

Figure 9.5-2 Serial output settings (when using external shift clock)

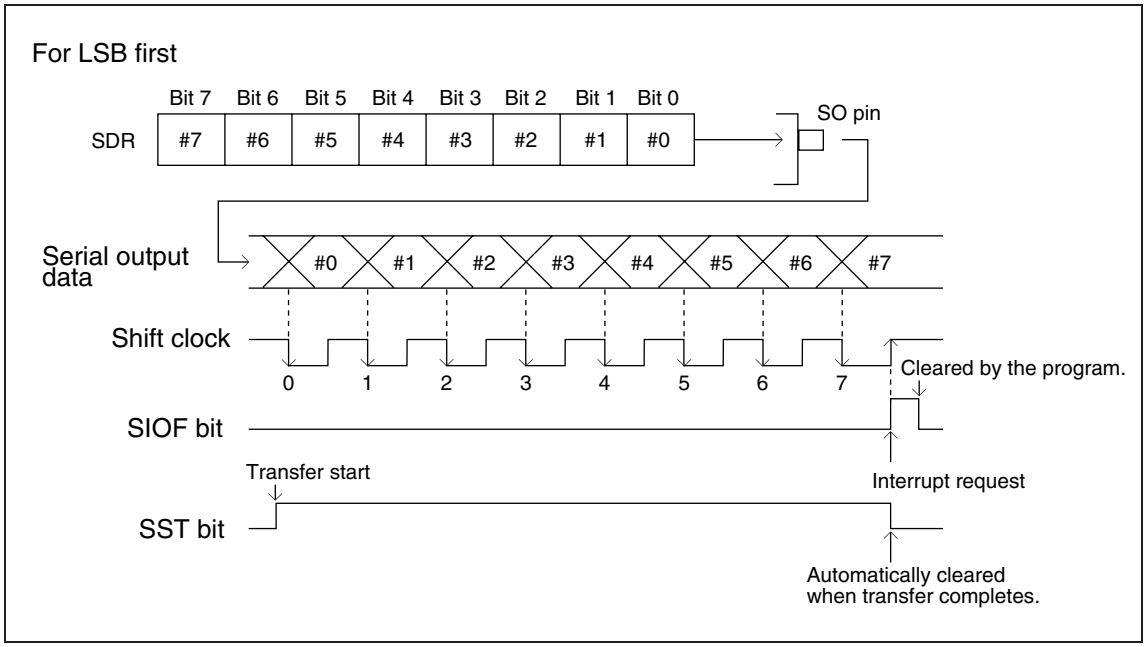


Enabling serial output operation outputs the contents of the SDR register to the SO pin, synchronized with the falling edge of the external shift clock. When serial output completes, reset the SDR register and enable operation (SMR: SST = "1") promptly for output of the next data.

When the device being communicated with has completed the serial input operation (on the rising edge), hold the external shift clock at the "H" level while waiting for next output data (idle state).

Figure 9.5-3 "8-bit serial output operation" shows the 8-bit serial output operation.

Figure 9.5-3 8-bit serial output operation



■ Operation at completion of serial output

The 8-bit serial I/O sets the interrupt request flag bit (SMR: SIOF = "1") and clears the serial I/O transfer start bit (SMR: SST = "0") on the rising edge of the shift clock after the serial data of the eighth bit is output.

9.6 Operation of Serial Input

The 8-bit serial I/O can perform serial input of 8-bit data synchronized with a shift clock.

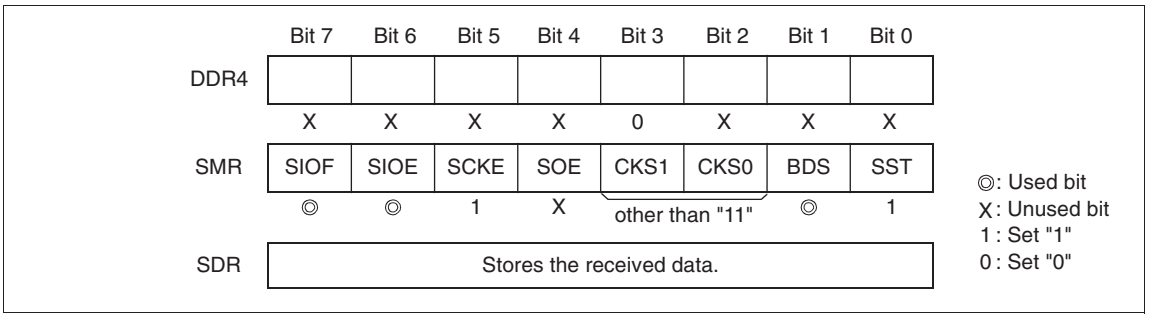
■ Serial input operation

Serial input can operate using an internal or external shift clock. When serial in operation is enabled, input from the serial data input pin (SI) is stored in SDR register. Serial output is performed at the same time.

● Internal shift clock

Figure 9.6-1 "Serial input settings (when using internal shift clock)" shows the settings required to operate serial input using an internal shift clock.

Figure 9.6-1 Serial input settings (when using internal shift clock)

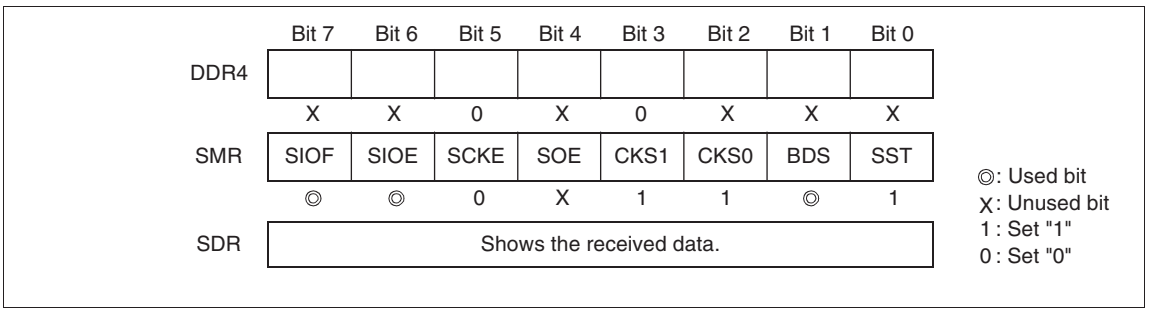


Starting serial input operation stores the value of the serial data input pin (SI) to the SDR register, synchronized with the rising edge of the selected internal shift clock. At this time, the device being communicated with (a serial output) must have data set in the SDR register and be waiting for input of the external shift clock.

● External shift clock

Figure 9.6-2 "Serial input settings (when using external shift clock)" shows the settings required to operate serial input using an external shift clock.

Figure 9.6-2 Serial input settings (when using external shift clock)

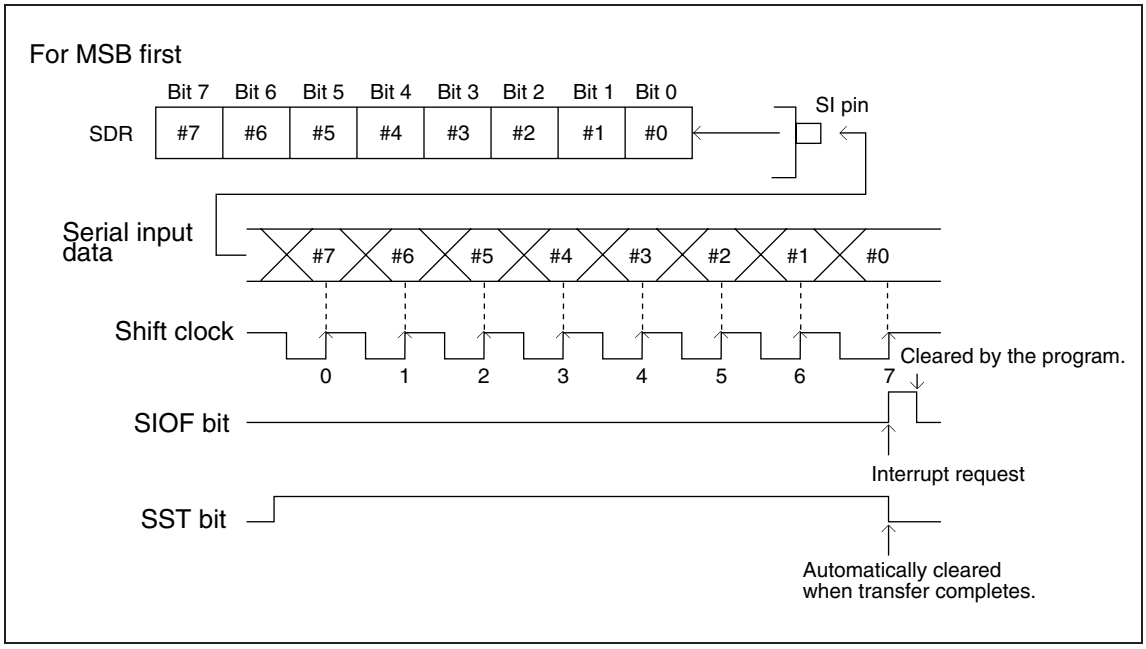


Enabling serial input operation stores the data on the SI pin to the SDR register, synchronized with the rising edge of the external shift clock. When serial input completes, read the SDR register and enable operation (SMR: SST = "1") promptly to input next data.

During this time, hold the external shift clock at the "H" level while waiting for the next data (idle state).

Figure 9.6-3 "8-bit serial input operation" shows the 8-bit serial input operation.

Figure 9.6-3 8-bit serial input operation



■ Operation at completion of serial input

The 8-bit serial I/O sets the interrupt request flag bit (SMR: SIOF = "1") and clears the serial I/O transfer start bit (SMR: SST = "0") on the rising edge of the shift clock after the serial data of the eighth bit is input.

9.7 States in Each Mode during 8-bit Serial I/O Operation

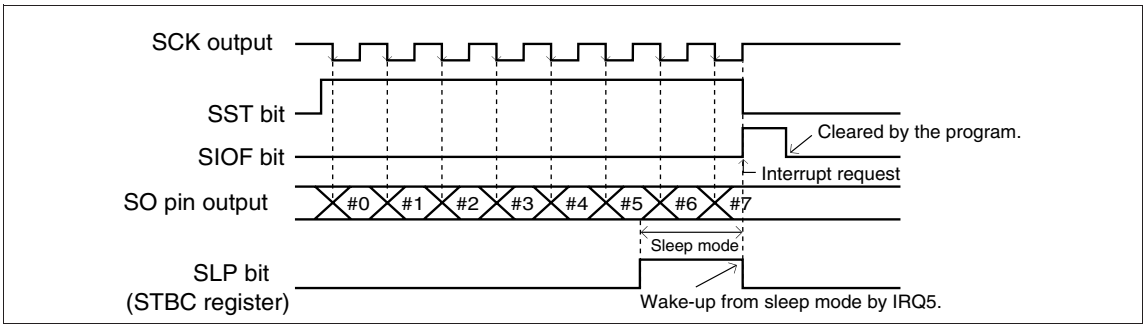
This section describes the operation of the 8-bit serial I/O when the device goes to sleep or stop mode, or an operation halt request occurs during transfer.

■ Using internal shift clock

● Operation in sleep mode

In sleep mode, serial I/O operation does not halt and transfer continues, as shown in Figure 9.7-1 "Operation in sleep mode (internal shift clock)".

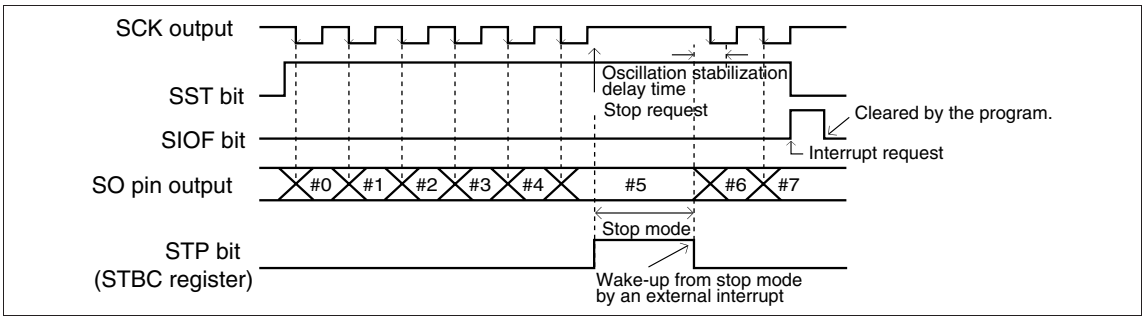
Figure 9.7-1 Operation in sleep mode (internal shift clock)



● Operation in stop mode

In stop mode, serial I/O operation and transfer stop, as shown in Figure 9.7-2 "Operation in stop mode halt (internal shift clock)". As operation restarts after wake-up from stop mode, initialize the 8-bit serial I/O depending on the state of the device with the 8-bit serial I/O is communicating.

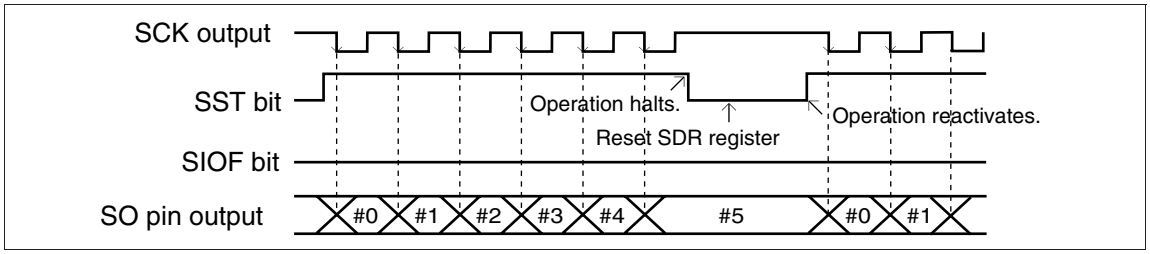
Figure 9.7-2 Operation in stop mode halt (internal shift clock)



● Operation during halt

Halting operation during transfer (SMR: SST = "0") halts the transfer and clears the shift clock counter, as shown in Figure 9.7-3 "Operation during halt (internal shift clock)". Therefore, the device being communicated with must also be initialized. In serial output operation, set data to the SDR register again before reactivating.

Figure 9.7-3 Operation during halt (internal shift clock)

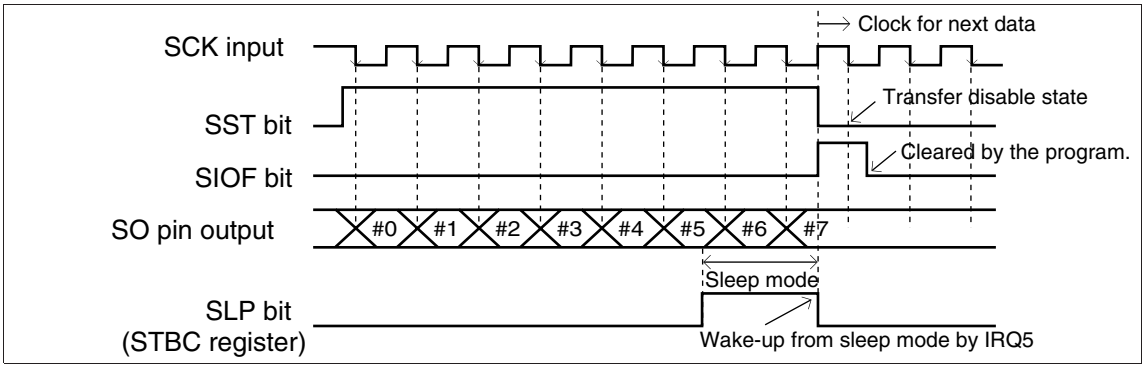


■ Using external shift clock

● Operation in sleep mode

In sleep mode, serial I/O operation does not halt and transfer continues, as shown in Figure 9.7-4 "Operation in sleep mode (external shift clock)".

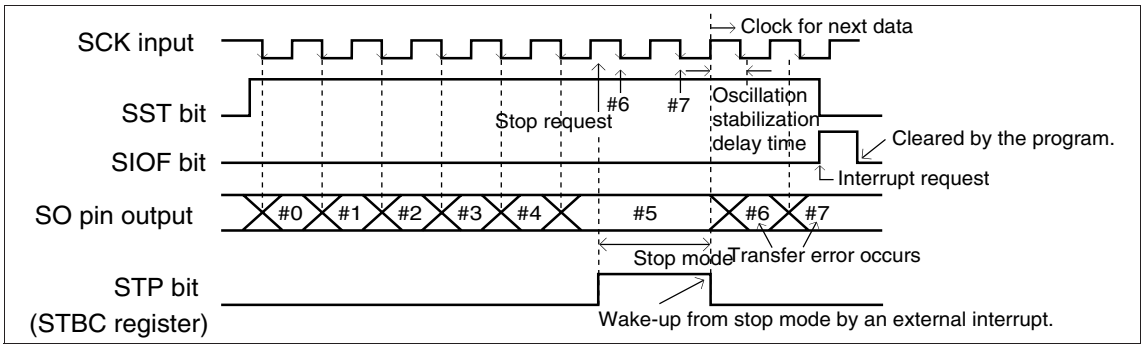
Figure 9.7-4 Operation in sleep mode (external shift clock)



● Operation in stop mode

In stop mode, serial I/O operation halts and transfer aborts, as shown in Figure 9.7-5 "Operation in stop mode (external shift clock)". Operation restarts after wake-up from stop mode. This causes an error to occur on the device with which the 8-bit serial I/O is communicating. Initialize the 8-bit serial I/O after wake-up from stop mode.

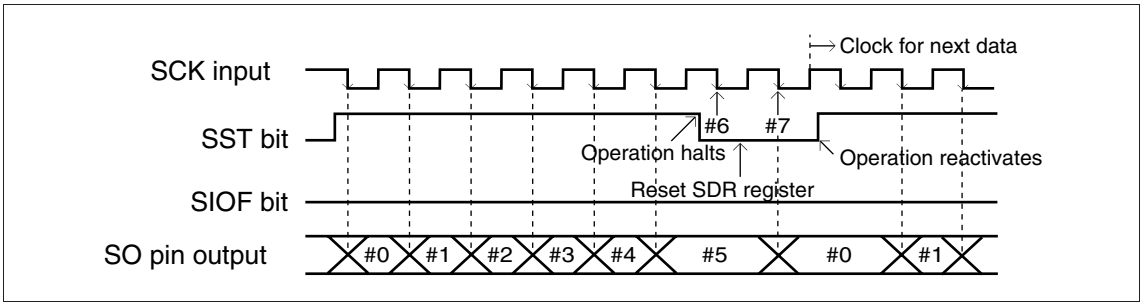
Figure 9.7-5 Operation in stop mode (external shift clock)



● Operation during halt

Halting operation during transfer (SMR: SST = "0") halts the transfer and clears the shift clock counter, as shown in Figure 9.7-6 "Operation during halt (external shift clock)". Therefore, the device being communicated with must also be initialized. In serial output operation, set the SDR register again before re-activating. If an external clock is input at this time, the SO pin output changes.

Figure 9.7-6 Operation during halt (external shift clock)



9.8 Notes on Using 8-bit Serial I/O

This section lists points to note using when the 8-bit serial I/O.

■ Notes on using 8-bit serial I/O

● Error on starting serial transfer

Activating the serial transfer by software (SMR: SST = "1") is not synchronized with the falling edge (output) or rising edge (input) of the shift clock, there is a delay of up to one cycle of the selected shift clock before the first serial data I/O occurs.

● Malfunction due to noise

In serial data transfer, malfunction of the serial I/O may occur if unwanted pulses (pulses exceeding the hysteresis width) occur on the shift clock due to external noise.

● Notes on setting by program

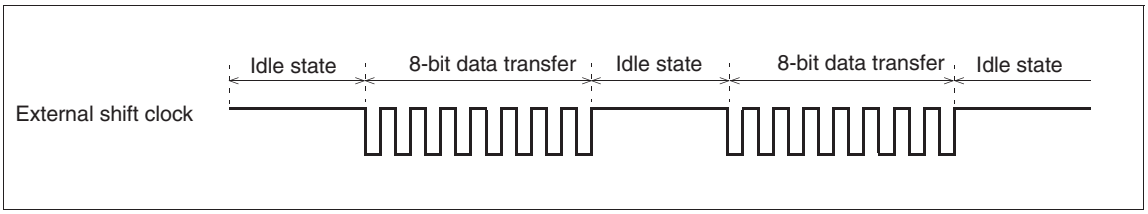
- Write to the serial mode register (SMR) and serial input register (SDR) when serial I/O is stopped (SMR: SST = "0").
- Do not modify other SMR register bits when starting/enabling serial I/O transfer (SMR: SST = "1").
- When using an external shift clock and when serial data output is enabled (SMR: SOE = "1"), the output level on the SO pin when the external shift clock is the most significant bit (when MSB first is selected) or least significant bit (when LSB first is selected). This applies even if serial transfer is stopped (SMR: SST = "0").
- The interrupt request flag bit (SMR: SIOF) is not set if serial I/O transfer is stopped (SMR: SST = "0") at the same time as serial transfer data completes.
- Interrupt processing cannot return if the SIOF bit is "1" and the interrupt requests enable bit is enabled (SIOE = "1"). Always clear the SIOF bit.

● Idle state of shift clock

Hold the external shift clock at the "H" level during the delay time between transfers of 8-bit data (idle state). When set as the shift clock output (SMR: SCKE = "1"), the internal shift clock (SMR: CKS1, CKS0 = other than "11_B") output an "H" level during the idle state.

Figure 9.8-1 "Idle state of shift clock " shows the idle state of the shift clock.

Figure 9.8-1 Idle state of shift clock



9.9 Connection Example for 8-bit Serial I/O

This section shows an example of connecting together two MB89950/950A series 8-bit serial I/O and performing bi-directional serial I/O.

■ Bi-directional serial I/O performing

Figure 9.9-1 Connection example for 8-bit serial I/O (interface between two MB89950/950A)

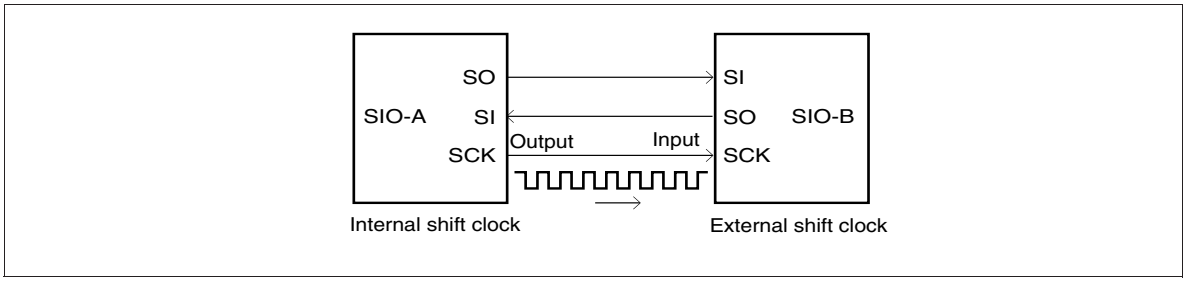
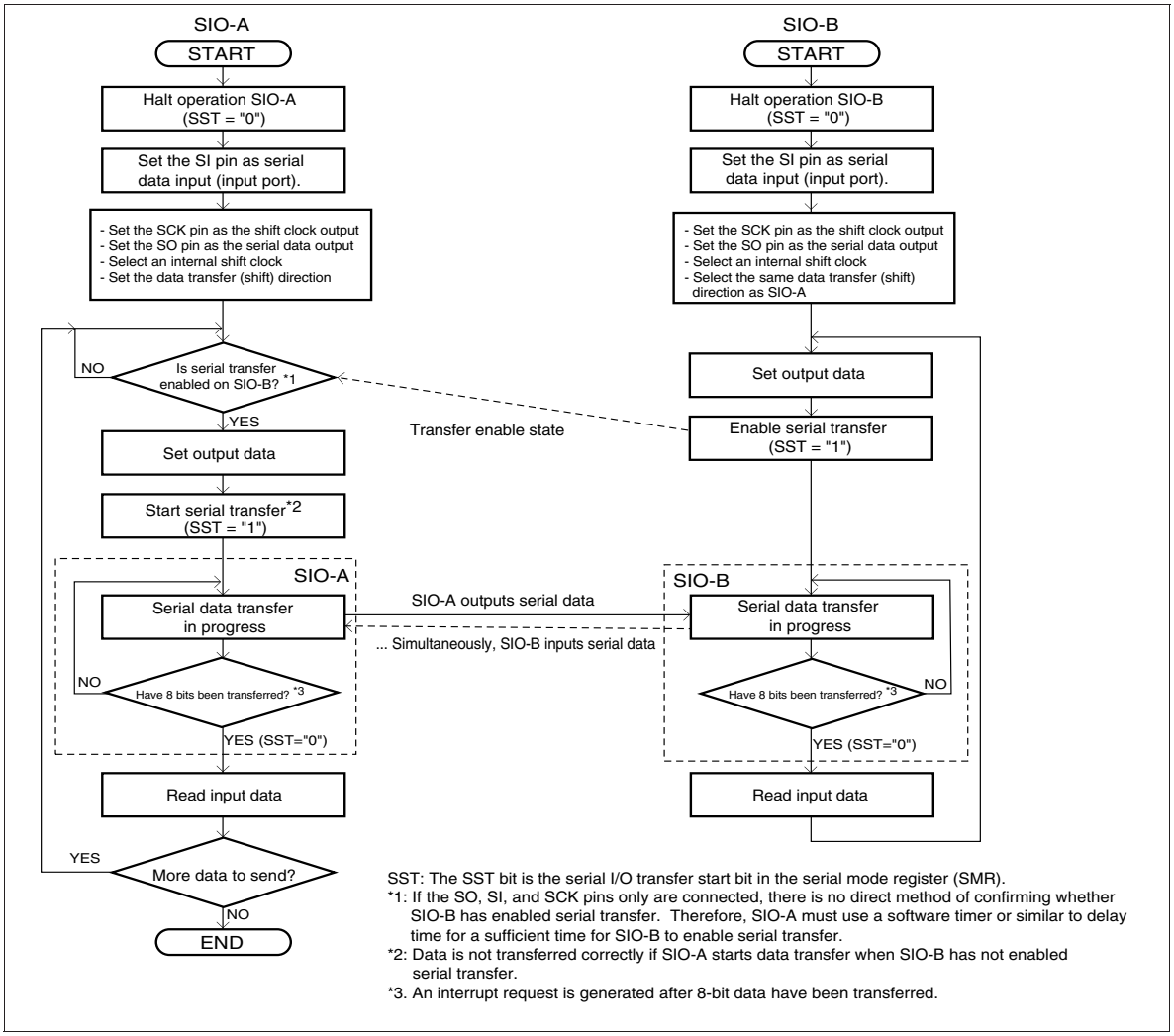


Figure 9.9-2 Operation of bi-directional serial I/O



9.10 Program Example for 8-bit Serial I/O

This section gives program example for 8-bit serial I/O.

■ Program example for serial output

● Processing description

- Outputs 8-bit serial data (55_H) from the SO pin of serial I/O, then generates an interrupt when transfer is completed.
- The interrupt processing routine resets the transfer data and continues output.
- Operates as an internal shift clock and outputs the shift clock from the SCK pin.
- With a main clock oscillation frequency F_{CH} of 5 MHz, the highest speed clock selected by the speed-shift function (1 instruction cycle = $4/F_{CH}$), and a 32 t_{inst} shift clock, the data transfer rate will be as follows.

 Transfer speed = 5 MHz/4/32 = 39 kbps

 Interrupt cycle = 8 x 32 x 4/5 MHz = 204.8 µs

● Coding example

```
SMR     EQU    001CH           ; Serial mode register
SDR     EQU    001DH           ; Serial data register

SIOF    EQU    SMR:7           ; Define the interrupt request flag bit.
SST     EQU    SMR:0           ; Define the serial I/O transfer start bit.

ILR2    EQU    007DH           ; Address of the interrupt level setting register 2

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF0H
IRQ5    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   SST             ; Stop serial I/O transfer.
        MOV    ILR2,#11110111B ; Set interrupt level (level 1).
        MOV    SDR,#55H        ; Set transfer data (55H).
        MOV    SMR,#01111000B  ; Clear Interrupt request flag, enable interrupt
                               ; request output, enable shift clock output (SCK),
                               ; enable serial data output (SO), select 32 tinst,
                               ; LSB first.
        SETB   SST             ; Start serial I/O transfer.
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   SIOF            ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T             ; Save A and T.
        PUSHW  A
        MOV    SDR,#55H        ; Reset transfer data (55H).
        SETB   SST             ; Start serial I/O transfer.
         :
        User processing
         :
        POPW   A
        XCHW   A,T             ; Restore A and T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

■ Program example for serial input

● Processing description

- Inputs 8-bit serial data from the SI pin of serial I/O, then generates an interrupt when transfer is completed.
- The interrupt processing routine reads the transferred data and continues transfer.
- Serial I/O uses the external shift clock. The shift clock is input from the SCK pin.

● Coding example

```
DDR4    EQU    000FH
SMR     EQU    001CH           ; Serial mode register
SDR     EQU    001DH           ; Serial data register

SIOF    EQU    SMR:7           ; Define the interrupt request flag bit.
SST     EQU    SMR:0           ; Define the serial I/O transfer start bit.

ILR2    EQU    007DH           ; Address of the interrupt level setting register 2

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF0H
IRQ5    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        MOV    DDR4,#00000000B ; Set P45/SCK and P43/SI pin as an input.
        CLRI                   ; Disable interrupts.
        CLRB   SST             ; Stop serial I/O transfer.
        MOV    ILR2,#11110111B ; Set interrupt level (level 1).
        MOV    SMR,#01001100B  ; Clear interrupt request flag, enable interrupt
                               ; request output, set shift clock input (SCK),
                               ; disable serial data output (SO), select the
                               ; external shift clock, LSB first.
        SETB   SST             ; Enable serial I/O transfer.
        SETI                   ; Enable interrupts.
         :
;-------------------------------------------------------------------------------
WARI    CLRB   SIOF            ; Clear interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
        MOV    A,SDR           ; Read transfer data.
        SETB   SST             ; Enable serial I/O transfer.
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;-------------------------------------------------------------------------------
        END
```

CHAPTER 10
UART

This chapter describes the functions and operation of the UART.

10.1 "Overview of UART"

10.2 "Structure of UART"

10.3 "UART Pins"

10.4 "UART Registers"

10.5 "UART Interrupts"

10.6 "Operation of UART"

10.7 "Operation of Mode 0, 1, 3"

10.8 "Program Example for UART"

10.1 Overview of UART

The UART is a general-purpose data communication interface. The UART supports both synchronous clock and asynchronous clock mode and transmits variable-length serial data. The transmission format is the "NRZ" system and the transmission data rate is configurable by setting the proprietary baud rate generator, external clocks, internal timers.

■ UART function

The UART communicates with other CPU's and peripheral devices by transmitting/receiving serial data (serial input/output).

- The full-duplex double buffer embedded in the device enables full-duplex bi-directional communication.
- User can configure the UART to the synchronous transfer mode or asynchronous transfer mode.
- Internal baud rate generator allows user to select a baud rate from eight different speed (for internal clock). The baud rate is also configured by setting external clock inputs and 8-bit PWM timer, allowing flexible setting of rate.
- The variable data length system allows users to set the data length at 5, 8 and 9 bit with non-parity or 4, 7 and 8 bit with parity (See Table 10.1-1 "UART operating mode").
- The data transmission format is based on the NRZ (Non Return to Zero) system.

Table 10.1-1 UART operating mode

Operating mode	Data length		Clock mode	Stop bit length
	Non-parity	Parity		
0	5	4	Asynchronous/Synchronous	1 bit or 2 bits (*1)
1	8	7	Asynchronous/Synchronous	1 bit or 2 bits (*1)
2	8+1	--	Asynchronous/Synchronous	1 bit or 2 bits (*1)
3	9	8	Asynchronous/Synchronous	1 bit or 2 bits (*1)

*1: In the receive mode, only the stop bit of length 1 is valid and the second bit received is always ignored.

■ Selection of transfer clocks

The transfer clock can selected from the external clock (SCK pin), PWM timer or dedicated baud rate generator by setting CS0 and CS1 bits of serial rate control register (SRC). In addition, the CR bit of SRC and SMDE bit of serial mode control register 1 (SMC1) can determine which divider for the selected transfer clock. Please refer to Table 10.1-2 "Clock ratio".

Table 10.1-2 Clock ratio

CS1	CS0	Clock input	CR	Asynchronous	Synchronous
0	0	External clock	0	1/16	1/1
			1	1/64	
0	1	PWM timer	0	1/16	1/2
			1	1/64	
1	0	Dedicated baud rate generator	0	1/16	1/2
			1	1/64	
1	1		--	1/8	1/1

When using the dedicated baud rate generator, the input clock of the baud rate generator is selected by PDS1 and PDS0 bits of serial mode control register 2 (SMC2). The ratio of dividing frequency is shown in Table 10.1-3 "Dividing frequency of dedicated baud rate generator".

Table 10.1-3 Dividing frequency of dedicated baud rate generator

PDS1	PDS0	Dividing frequency	Input clock
0	0	1/4	CPU operating clock
0	1	1/6	CPU operating clock
1	0	1/13	CPU operating clock
1	1	1/65	CPU operating clock

Table 10.1-4 "Transfer cycle and transfer rate by baud rate generator" is shown the example of baud rate when using the dedicated baud rate generator.

Table 10.1-4 Transfer cycle and transfer rate by baud rate generator

RC2	RC1	RC0	Division ratio	Baud rate (bps)			
				4.912 MHz	5 MHz		Input clock
				1/4	1/4	1/65	PDS division
				1/64	1/8	1/16	CS1, CS0, CR division
0	0	0	20	9600	78125	2404	
0	0	1	21	4800	39063	1202	
0	1	0	22	2400	19531	601	
0	1	1	23	1200	9766	300	
1	0	0	24	600	4883	150	
1	0	1	25	300	2441	75	
1	1	0	26	150	1221	38	
1	1	1	27	75	610	19	

Furthermore, Figure 10.1-1 "Sample calculation of the baud rate" shows the formula of calculating the baud rate.

Figure 10.1-1 Sample calculation of the baud rate

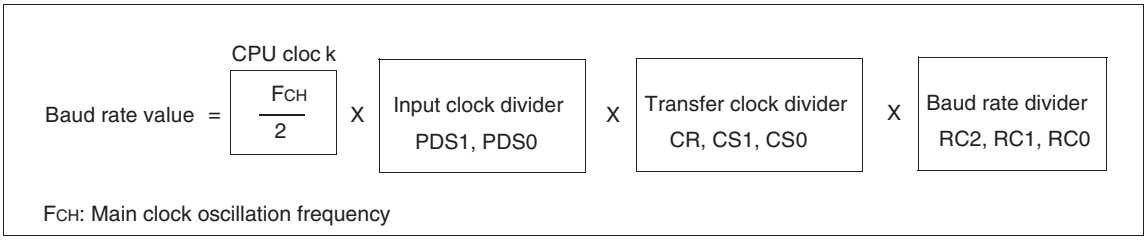


Table 10.1-5 Transfer cycle and transfer rate by external clocks

Asynchronous transfer mode				Synchronous transfer mode		
Selected baud rate division value		Transfer cycle	Transfer rate (baud) (*1)	Selected baud rate division value	Transfer cycle	Transfer rate (baud) (*1)
CR = 0	16	$128/F_{CH}$ or more	39062 or less	1	$8/F_{CH}$ or more	625k or less
CR = 1	64	$512/F_{CH}$ or more	9765 or less			

F_{CH}: Main clock oscillation frequency

*1: Min. external clock cycle of ($8/F_{CH}$ = 0.16 µs) for F_{CH} set at 5 MHz.

Figure 10.1-2 Sample calculation of the baud rate (external clock is selected)

$$\text{Baud rate value} = \frac{\text{External clock input } (F_{CH}/2/4 \text{ min.})}{CR \begin{pmatrix} CR=0...16 \\ CR=1...64 \end{pmatrix}}$$

FCH: Main clock oscillation frequency

Table 10.1-6 Transfer cycle and transfer rate by 8-bit PWM timers

PWM timer count clock cycle		Asynchronous transfer mode			Synchronous transfer mode	
		Clock division value		Transfer rate (baud) (*1)	Clock division value	Transfer rate (baud) (*1)
1 t_{inst}		CR = 0	16	39062 to 152.6	2	312.5k to 1.22k
		CR = 1	64	9765.6 to 38.1		
16 t_{inst}		CR = 0	16	2441.4 to 9.5	2	19531.3 to 76.3
		CR = 1	64	610.4 to 2.4		
64 t_{inst}		CR = 0	16	610.4 to 2.4	2	4882.8 to 19.1
		CR = 1	64	152.6 to 0.6		
From PWC timer	1 t_{inst}	CR = 0	16	19531.3 to 76.3	2	156.3 to 610.4
		CR = 1	64	4882.8 to 19.1		
	4 t_{inst}	CR = 0	16	4882.8 to 19.1	2	39062 to 152.6
		CR = 1	64	1220.7 to 4.8		
	32 t_{inst}	CR = 0	16	610.4 to 2.38	2	4882.8 to 19.1
		CR = 1	64	152.3 to 0.6		

t_{inst}: Instruction cycle
*1: Main clock oscillation frequency (F_{CH}) = 5 MHz

Figure 10.1-3 Sample calculation of the baud rate (PWM timer is selected)

$$\frac{1}{\text{Baud rate value}} = \text{Clock selection } [4/F_{CH}] \times \text{Input clock select bits (PWM timer) } [1\ (P1=0, P0=0)\ /\ 16\ (P1=0, P0=1)\ /\ 32\ (P1=1, P0=0)\ /\ PWC\ (P1=1, P0=1)^*] \times \text{Compare register (COMR) } [\text{Compare register value}+1] \times 2 \times CR\ (\text{Cycle time}) \left(\begin{array}{l} CR = 0 \ldots 16 \\ CR = 1 \ldots 64 \end{array} \right)$$

F_{CH}: Main clock oscillation frequency
* : PWC value depends on the value of PWC pulse width control register 2, bit 3 and bit 2 (PCR2:C1, C0).
C1=0, C0=0 -> 1
C1=0, C0=1 -> 4
C1=1, C0=0 -> 32
C1=1, C0=1 -> Prohibited

Refer to CHAPTER 7 "8-BIT PWM TIMER" section for information on the count clock cycle of the PWM timer, PWM compare register setting value and output cycle of the PWM timer.

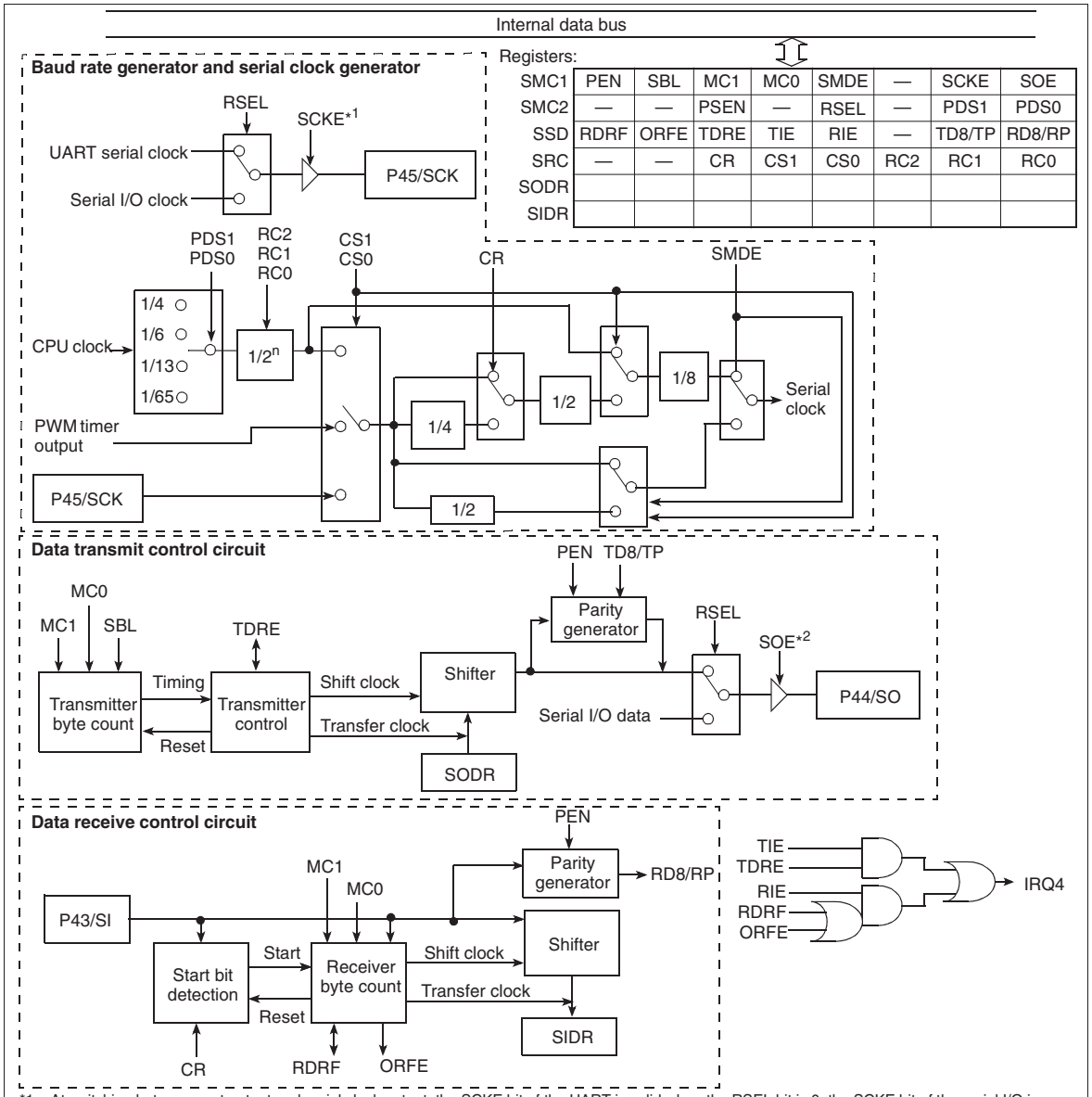
10.2 Structure of UART

The UART consists of the following blocks:

- **Baud rate generator and serial clock generator**
- **Data transmitter and data receiver**
- **Registers (SMC1, SMC2, SRC, SSD, SIDR, SODR)**

■ Block diagram of UART

Figure 10.2-1 Block diagram of UART



*1: At switching between port output and serial clock output, the SCKE bit of the UART is valid when the RSEL bit is 0; the SCKE bit of the serial I/O is valid when the RSEL bit is 1.

*2: At switching between port output and serial data output, the SOE bit of the UART is valid when the RSEL bit is 0; the SOE bit of the serial I/O is valid when the RSEL bit is 1.

- Baud rate generator and serial clock generator

 This block generates transmit/receive clocks from the outputs of baud rate generator, 8-bit PWM timer or external clock.

- Date receive control circuit

 The receive control circuit consists of the receive byte counter, the start bit detection circuit and the receive parity circuit.

 The receive byte counter counts number of data bit received and generates an interrupt after having received data of the specified length.

 The start bit detection circuit detects start bit from the serial input pin and starts to shift the following data bit received into the shifter.

 The receive parity circuit stores a parity bit after receiving data with a parity. When 9-bit long data is received, the receive parity circuit stores the last bit received.

- Data transmit control circuit

 The transmit control circuit consists of a transmission byte counter and a transmission parity circuit.

 The transmit byte counter counts number of data bytes transmit and generates an interrupt after having received a data of the set length.

 The transmission circuit generates a parity bit when transmitting data with a parity bit. When 9-bit long data is sent, the MSB of the transmit data is sent.

- Serial mode control register 1 (SMC1)

 This register controls operating modes in the UART. The register is used to select parity/non-parity, stop bit length, operating mode (data length), synchronous/asynchronous, enable/disable of UART serial clock output (SCK) and enable/disable of serial data output (SO).

- Serial mode control register 2 (SMC2)

 This register controls UART starting/stopping operation, UART/SIO function and input clock divider of the baud rate generator.

- Serial rate control register (SRC)

 This register controls the data transmission rate (baud rate). The register selects transfer rate generated by the baud rate generator.

- Serial status and data register (SSD)

 This register is used to select or show transmit/receive operation, to indicate error status, and to select received/transmitted data parity.

- Serial input data register (SIDR)

 This register holds received data. Serial data received is converted to parallel data and stored in the register. When the data length is set to 7 bits, bit 7 does not have meaning.

● Serial output data register (SODR)

This register stores data to be transmitted. The data written in this register is converted to serial data and sent to serial output pin. When the data length is set to be 7 bits, bit 7 does not have meaning.

10.3 UART Pins

This section describes the pins and pin block diagram of UART.

■ UART pins

The pins for the UART function are shift clock input/output pin (P45/SCK), serial data output pin (P44/SO) and serial data input pin (P43/SI).

P45/SCK:

This pin function either as a general-purpose input/output port (P45) or a clock input output pin (hysteresis input) for the UART(SCK).

When clock output is enabled (SMC1: SCKE = "1"), this pin functions as clock output pin (SCK) irrespective of settings on corresponding port direction register. In this case, do not select an external clock (SRC: CS1, CS0 are not "00_B").

To use the port as a UART clock input pin, disable the clock output (SMC1: SCKE = "0") and configure the port as an input port by setting a corresponding port direction register bit (DDR4: bit 5 = "0"). In this case, be sure to select an external clock (SRC: CS1, CS0 = "00_B").

P44/SO:

This pin functions either as general-purpose input/output port (P44) or serial data output pin of the UART (SO).

When serial data output is enabled (SMC1: SOE = "1"), this pin functions as serial data output pin of the UART irrespective of settings on corresponding port direction register.

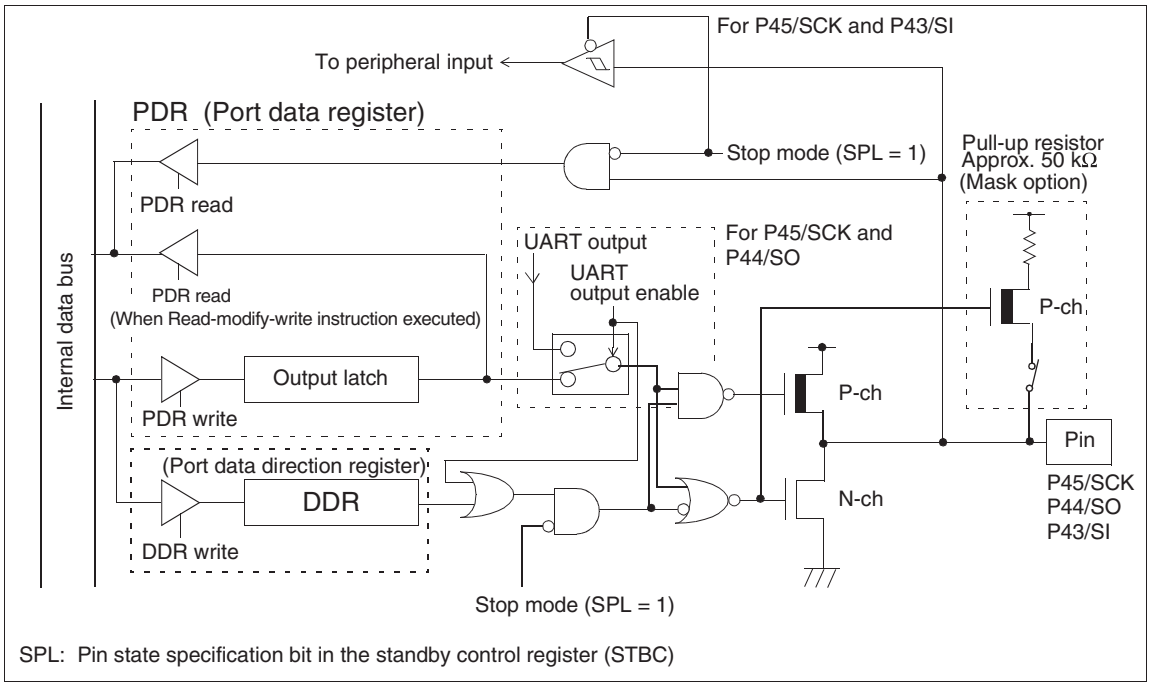
P43/SI:

This pin functions either as general-purpose input/output port (P43) or serial data input pin of the UART (SI).

To use the port as a UART serial data input pin, configure the port as output port by setting a corresponding bit of the port data direction register (DDR4: bit 3 = "0").

■ Block diagram of UART pins

Figure 10.3-1 Block diagram of UART pins



SPL: Pin state specification bit in the standby control register (STBC)

Reference:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

10.4 UART Registers

This section describes the registers of the UART.

■ UART registers

Figure 10.4-1 UART registers

SMC1 (Serial mode control register 1)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0020H	PEN	SBL	MC1	MC0	SMDE	—	SCKE	SOE	00000-00B
	R/W	R/W	R/W	R/W	R/W		R/W	R/W	

SRC (Serial rate control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0021H	—	—	CR	CS1	CS0	RC2	RC1	RC0	--011000B
			R/W	R/W	R/W	R/W	R/W	R/W	

SSD (Serial status and data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0022H	RDRF	ORFE	TDRE	TIE	RIE	—	TD8/TP	RD8/RP	00100-1XB
	R	R	R	R/W	R/W		R/W	R	

SIDR (Serial input data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0023H									XXXXXXXXB
	R	R	R	R	R	R	R	R	

SODR (Serial output data register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0023H									XXXXXXXXB
	W	W	W	W	W	W	W	W	

SMC2 (Serial mode control register 2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0024H	—	—	PSEN	—	RSEL	—	PDS1	PDS0	--1-0-00B
			R/W		R/W		R/W	R/W	

R/W : Readable and writable
R : Read-only
W : Write-only
— : Unused
X : Indeterminate

10.4.1 Serial Mode Control Register 1 (SMC1)

Serial mode control register 1 (SMC1) sets synchronous mode, stop bit length, data length, parity/non-parity and select the port function of SCK and SO.

■ Serial mode control register 1 (SMC1)

Figure 10.4-2 Serial mode control register 1 (SMC1)

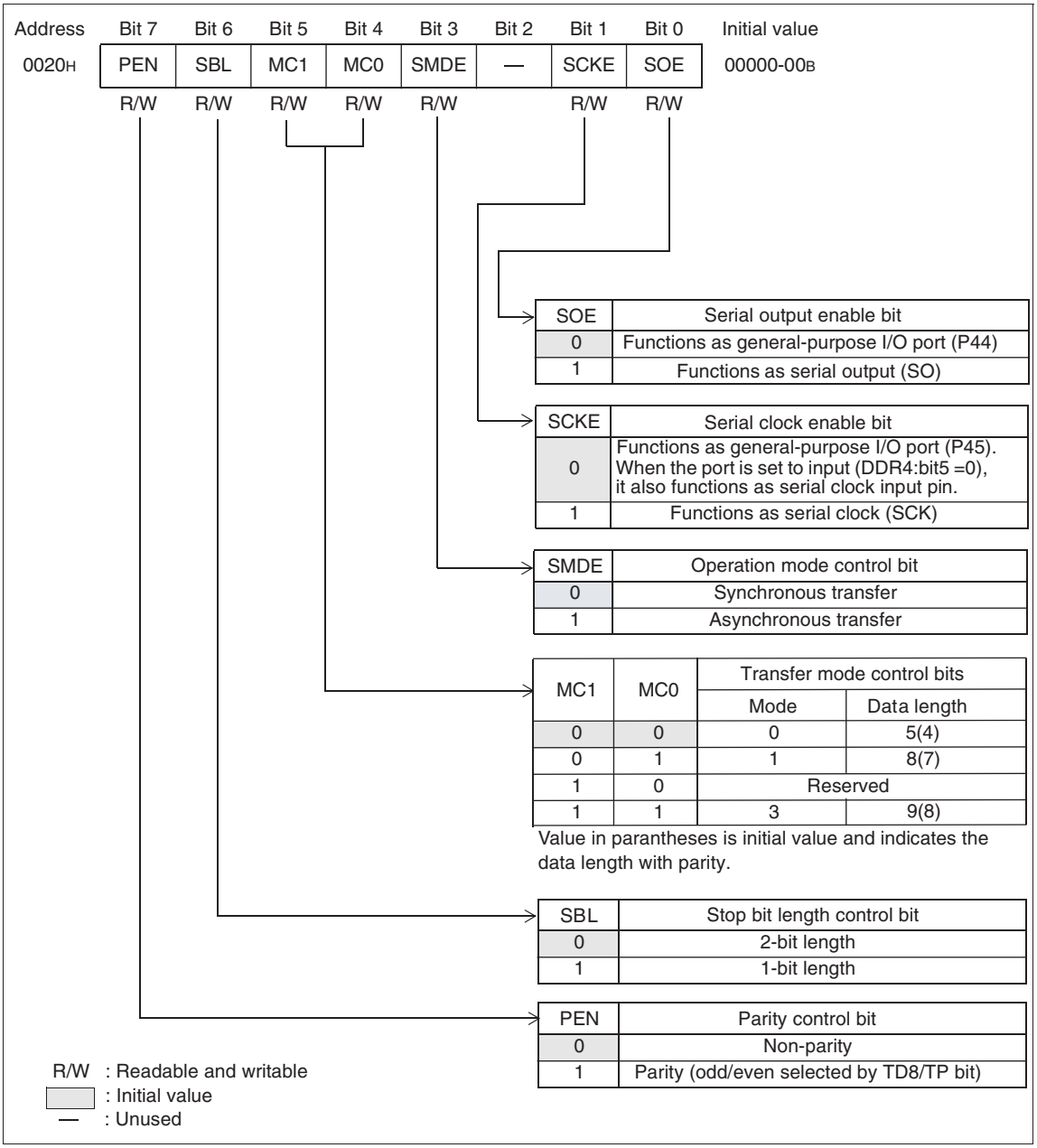


Table 10.4-1 Serial mode control register 1 (SMC1) bits

Bit		Function
Bit 7	PEN: Parity control bit	• In the clock asynchronous mode, sets whether there is parity data or not.
Bit 6	SBL: Stop bit length control bit	• This bit determines the stop bit length. • In serial transmission, a stop bit of the bit length specified is appended. • In serial reception, a stop bit is recognized as in a 1-bit length regardless of the value set here.
Bit 5 Bit 4	MC1, MC0: Transfer mode control bits	• These two bits determine the transfer mode (data length).
Bit 3	SMDE: Operation mode control bit	• This bit selects the UART operating mode. In asynchronous mode, the UART operates on the serial clock divided by 8. In clock synchronous mode, it operates on the selected serial clock.
Bit 2	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 1	SCKE: Serial clock output bit	• This bit selects either serial clock input/output (SCK) of the serial clock synchronous mode or general-purpose I/O port (P45). • When SCKE = "0" and the DDR4: bit 5 = "0", the SCK functions as serial clock input.
Bit 0	SOE: Serial data output bit	• This bit selects either serial data output (SO) or general-purpose I/O port (P44).

10.4.2 Serial Rate Control Register (SRC)

The serial rate control register (SRC) is to set the UART transmission speed (baud rate).

■ Serial rate control register (SRC)

Figure 10.4-3 Serial rate control register (SRC)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0021H	—	—	CR	CS1	CS0	RC2	RC1	RC0	--011000B
			R/W	R/W	R/W	R/W	R/W	R/W	

RC2	RC1	RC0	Baud rate selection bits: Division ratio	Baud rate (bps): 4.9152 MHz, 1/4	5 MHz, 1/4	5 MHz, 1/65	Clock: PDS division		
							CS1, CS0, CR division: 1/64	1/8	1/16
0	0	0	2^0	9600	78125	2404			
0	0	1	2^1	4800	39063	1202			
0	1	0	2^2	2400	19531	601			
0	1	1	2^3	1200	9766	300			
1	0	0	2^4	600	4883	150			
1	0	1	2^5	300	2441	75			
1	1	0	2^6	150	1221	38			
1	1	1	2^7	75	610	19			

CS1	CS0	Transfer clock selection bits: Clock input	CR	Asynchronous	Synchronous
0	0	External clock	0	1/16	1/1
			1	1/64	
0	1	PWM timer	0	1/16	1/2
			1	1/64	
1	0	Dedicated baud rate generator	0	1/16	1/2
			1	1/64	
1	1		—	1/8	1/1

CR	Clock rate selection bit
0	1/16 of clock input
1	1/64 of clock input

R/W : Readable and writable
: Initial value
— : Unused

Note: When CS1 and CS0 = "11B", the 1/8 clock rate is selected irrespective of the value of the CR bit.

Table 10.4-2 Serial rate control register (SRC) bits

Bit		Function
Bit 7 Bit 6	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 5	CR: Clock rate selection bit	• Used to select the asynchronous transfer clock rate. However, when the CS1 and CS0 bit are "11_B", the 1/8 clock rate is selected in spite of the value of the CR bit.
Bit 4 Bit 3	CS1, CS0: Transfer clock selection bits	• Used to select the clock input of the UART. If the external or internal clock is selected as clock input, the baud rate is a 1/16 or 1/64 clock frequency according to the value of the CR bit.
Bit 2 Bit 1 Bit 0	RC2, RC1, RC0: Baud rate selection bits	• Used to select the dedicated baud rate for the serial clock. One of eight baud rates can be selected from different combination of these bits.

10.4.3 Serial Status and Data Register (SSD)

The serial status and data register (SSD) is used to set and monitor transmit/receive operation and error status.

■ Serial status and rate register (SSD)

Figure 10.4-4 Serial status and data register (SSD)

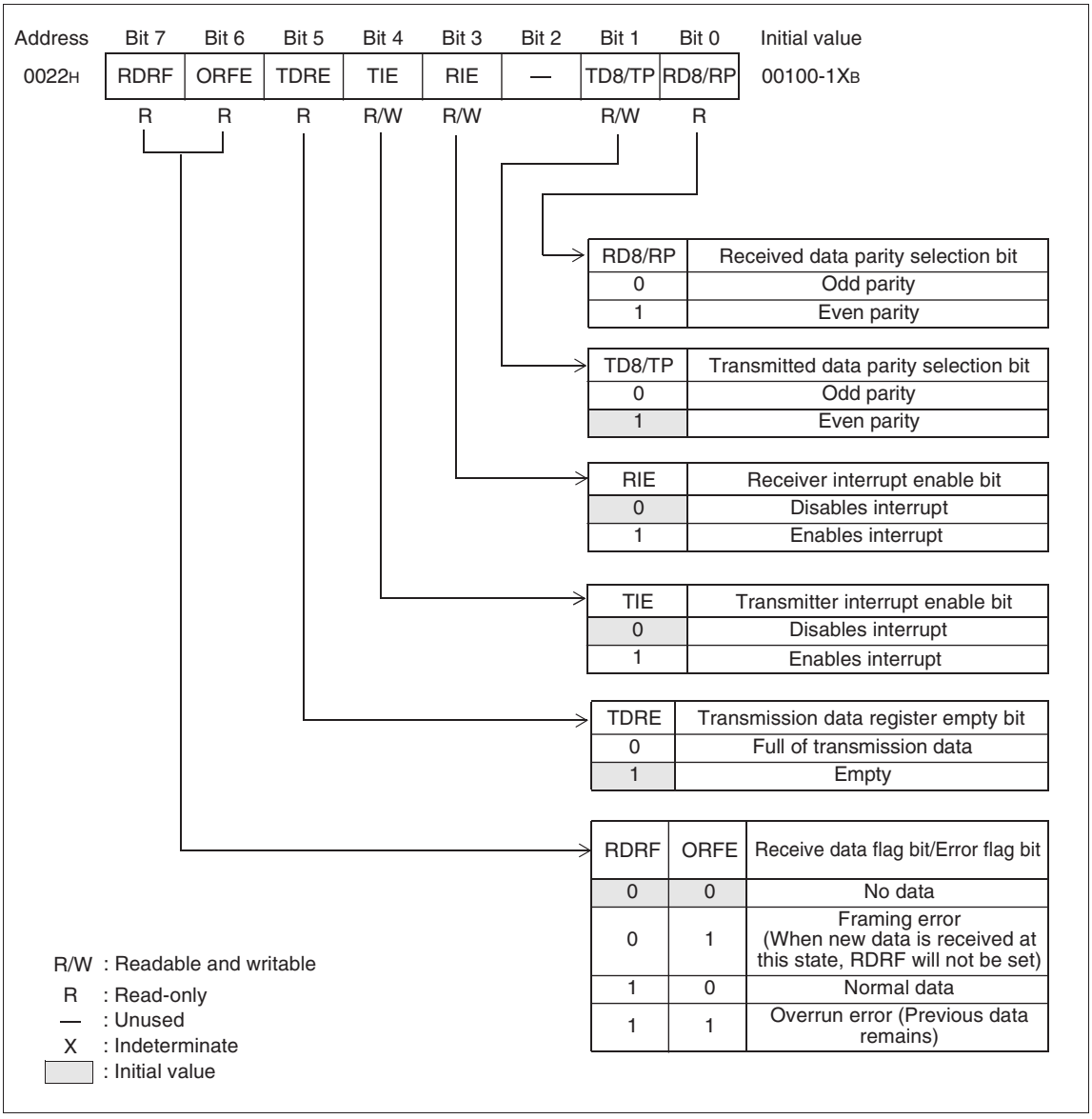


Table 10.4-3 Serial status and data register (SSD) bits

Bit		Function
Bit 7	RDRF: Receive data register full bit	• This flag represents the status of the serial input data register (SIDR). • This flag is set when receiving data is loaded into the SIDR register. It is cleared when the SIDR register is read. If the RDRF bit is set when the RIE bit is "1", a receive interrupt request is generated.
Bit 6	ORFE: Overrun/Framing error flag bit	• This bit is set when overrun or framing error is generated during receiving. • If this flag is set, data is not transferred from the receive shift register to SIDR register. • When the SIDR register is read after reading the SSD register with the ORFE flag set to "1", the ORFE flag is cleared to "0". • If the ORFE bit is set when the RIE bit is "1", a receive interrupt request is generated.
Bit 5	TDRE: Transmission data register empty bit	• This flag represents the status of the serial output data register (SODR). • This flag is cleared when transmission data is written into the SODR register. It is set when the data is loaded into the transmit shifter and transmission begins. If the TDRE bit is set when the TIE bit is "1", a transmission interrupt request is generated
Bit 4	TIE: Transmitter interrupt enable bit	• This bit enables transmission interrupt. If the TDRE bit is "1", a transmission interrupt is immediately generated once transmission interrupt enable bit is set to "1".
Bit 3	RIE: Receiver interrupt enable bit	• This bit enables receive interrupt. If the RDRF bit is "1" or if any error flag is "1", a receive interrupt is immediately generated once receive interrupt enable bit is set to "1".
Bit 2	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 1	TD8/TP: Transmitted data parity selection bit	• This bit is used to select the parity for the transmitted data.
Bit 0	RD8/RP: Received data parity selection bit	• This bit is used to select the parity for the received data.

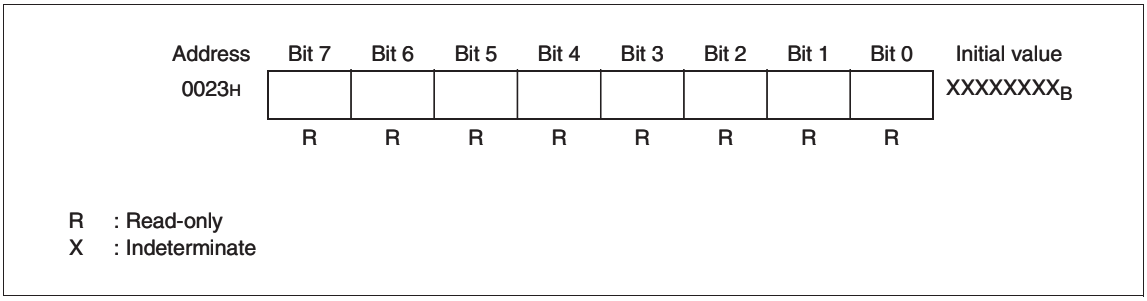
10.4.4 Serial Input Data Register (SIDR)

The serial input data register (SIDR) is used to input (receive) serial data.

■ Serial input data register (SIDR)

Figure 10.4-5 "Serial input data register (SIDR)" shows the bit allocations of the serial input data register.

Figure 10.4-5 Serial input data register (SIDR)



This register stores received data. Serial data received from the serial data input pin is converted to parallel in the shift register and stored in this register.

● Operation in mode 0 and 1

If received data is normally set in this register, the receive data flag bit (RDRF) is set to "1", and a receive interrupt request occurs if it is enabled. When the interrupt request is detected, check the RDRF bit in an interrupt processing or in a program. If there is receive data stored in this register, read this register, and then the RDRF flag is cleared automatically.

10.4.5 Serial Output Data Register (SODR)

The serial output data register (SODR) is used to output (transmit) serial data.

■ Serial output data register (SODR)

Figure 10.4-6 shows the bit allocations of the serial output data register.

Figure 10.4-6 Serial output data register (SODR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0023H									$XXXXXXXX_B$
	W	W	W	W	W	W	W	W	

W : Write-only
X : Indeterminate

When transmission is enabled, writing transmit data to this register transfers the transmit data to the transmit register. The transmit data is converted to serial in the transmit shift register and sent to the serial data output pin (SO).

Writing transmit data to the SODR register sets the transmit data flag to "0". After the transmit data is transferred to the transmit shift register, the transmit data flag is set to "1" and the SODR is ready for the next data. If transmit interrupt request is enabled, interrupt occurs. Write next transmission data when transmit data flag bit is set to "1". When the data length is set to 7 bits, bit 7 does not have meaning.

10.4.6 Serial Mode Control Register 2 (SMC2)

Serial mode control register 2 (SMC2) selects the division ratio of the baud rate generator, selects to function as UART or SIO, and enables the baud rate generator.

■ Serial mode control register 2 (SMC2)

Figure 10.4-7 Serial mode control register 2 (SMC2)

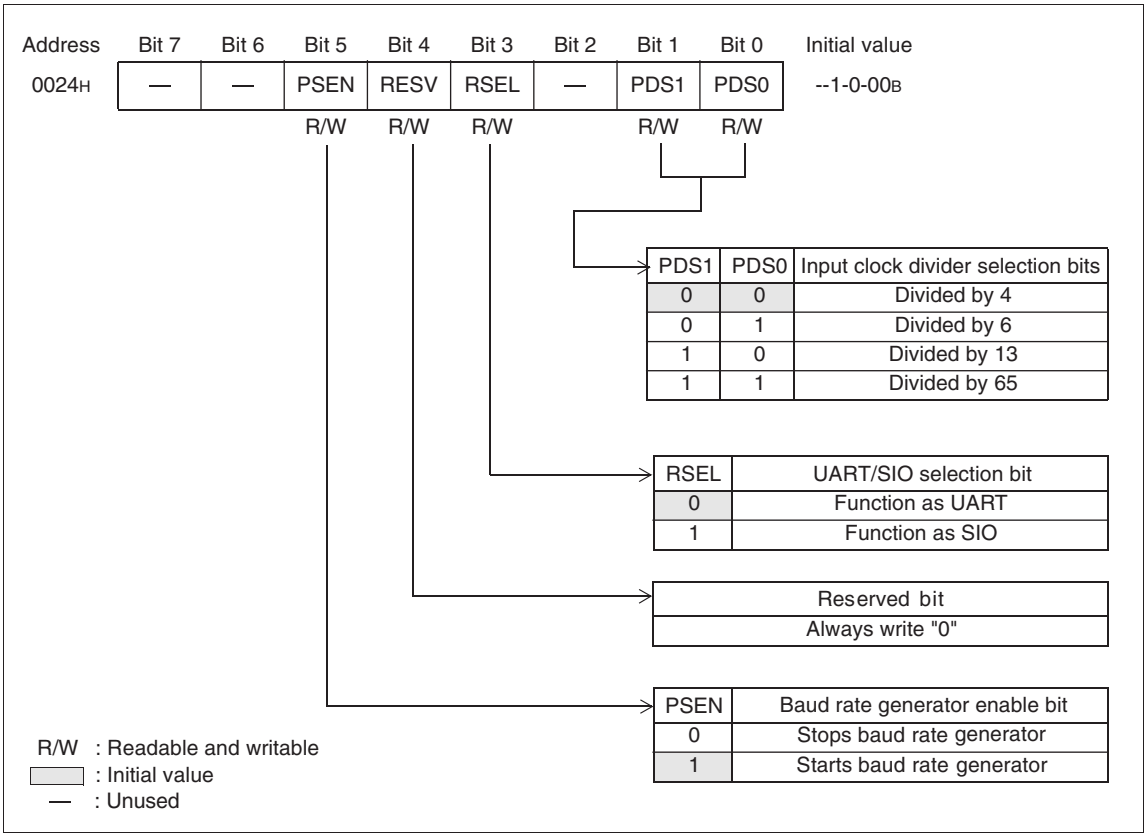


Table 10.4-4 Serial mode control register 2 (SMC2) bits

Bit		Function
Bit 7 Bit 6	Unused bits	• The read value is indeterminate. • Writing to these bits has no effect on the operation.
Bit 5	PSEN: Operation enable bit	• This bit enables baud rate generator. Baud rate generator is stopped by writing "0" to this bit after transmitting/receiving the current serial data, then disabled thereafter.
Bit 4	Reserved bit	• Always write "0".
Bit 3	RSEL: UART/SIO selection bit	• The bit is used to select whether the UART or serial I/O uses the data and clock I/O pins.
Bit 2	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 1 Bit 0	PDS1, PDS0: Input clock divider selection bits	• These bits are used to select the clock prescaler of the baud rate generator.

10.5 UART Interrupts

The UART has three interrupt causes -- transfer error interrupt, receive data full interrupt and transmit data empty interrupt:

- **When receive data is transferred from the receive shift register to the serial input data register (SIDR) (receive interrupt)**
- **When transmit data is transferred from the serial output data register (SODR) to the transmit shift register (transmit interrupt)**

■ Transmit interrupt

When transmission is enabled, writing transmit data to SODR register transfers the transmit data to the transmit register. The transmit data is converted to serial in the transmit shift register and sent to the serial data transmit pin (SO).

When the UART is ready to accept next data, the TDRE is set to "1", and an interrupt request (IRQ4) to the CPU is generated if transmit interrupt request is enabled (SSD: TIE = "1").

■ Receive interrupt

When the data is received normally (stop bit is detected), the RDRF is set to "1".

When an overrun error, a framing error or a parity error occurs, their corresponding error flag bit is set to "1". These bits are set when the stop bit(s) are detected, and an interrupt request (IRQ4) to the CPU is generated if receive interrupt is enabled (SSD: RIE = "1").

■ Registers and vector tables for UART interrupts

Table 10.5-1 Registers and vector tables for UART interrupts

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting bits		Upper	Lower
IRQ4	ILR2 ($007D_H$)	L41 (Bit 1)	L40 (Bit 0)	$FFF2_H$	$FFF3_H$

See Section 3.4.2 "Interrupt Processing" for details on the interrupt operation.

10.6 Operation of UART

This section describes the operation of the UART.
The UART has a serial communication function (operation mode 0,1,3).

■ Operation of UART

● Operation mode

The UART has 3 operation modes. The mode 0, 1, 3 are standard serial transmission modes in which a data type from 4-bit data length/parity to 9-bit data length/non-parity is selected (See Table 10.1-1 "UART operating mode").

● Transfer data format

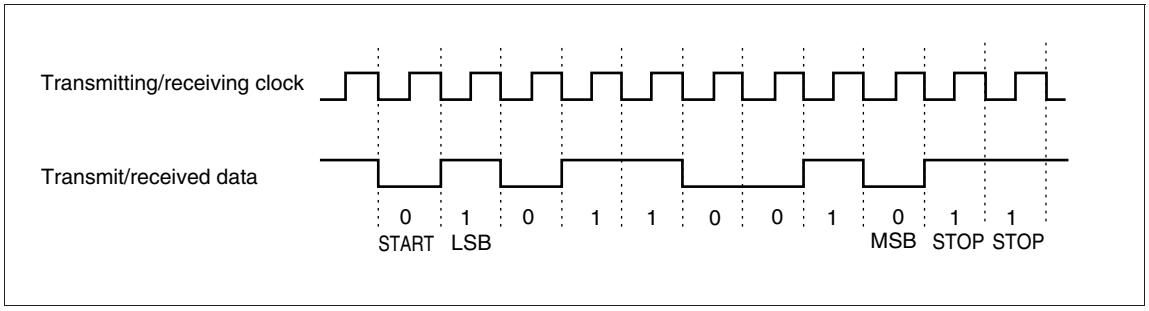
The UART can handle data type of the NRZ (Non Return to Zero) system only.

The transmission data always begins with a start bit ("L" level) followed by a specified length of data bits arranged in the "LSB first" format and ends with stop bit(s) ("H" level).

In asynchronous transfer mode, the relation between serial clock and serial input/output signal is not as shown in Figure 10.6-1 "Transfer data format".

Figure 10.6-1 "Transfer data format" shows the relation between transmit/receive clock and data in operation mode 1 when non-parity, 2 stop bits, synchronous transfer, transmit data of "01001101_B" (8 bits) are selected.

Figure 10.6-1 Transfer data format



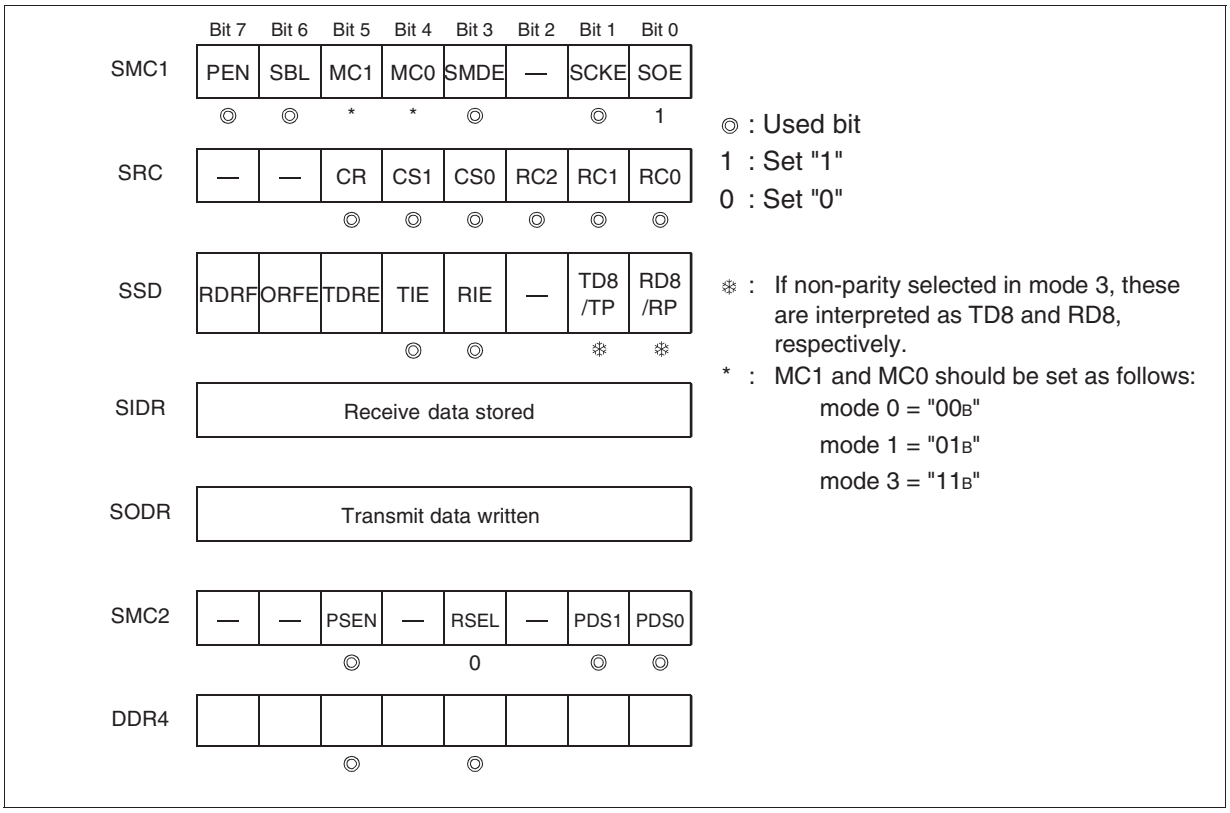
10.7 Operation of Mode 0, 1, 3

The operation mode 0, 1 and 3 provide a serial communication function.

■ Operation of operation mode 0, 1, 3

Settings shown in Figure 10.7-1 "Operation of operation mode 0, 1, 3" are necessary for the UART operation.

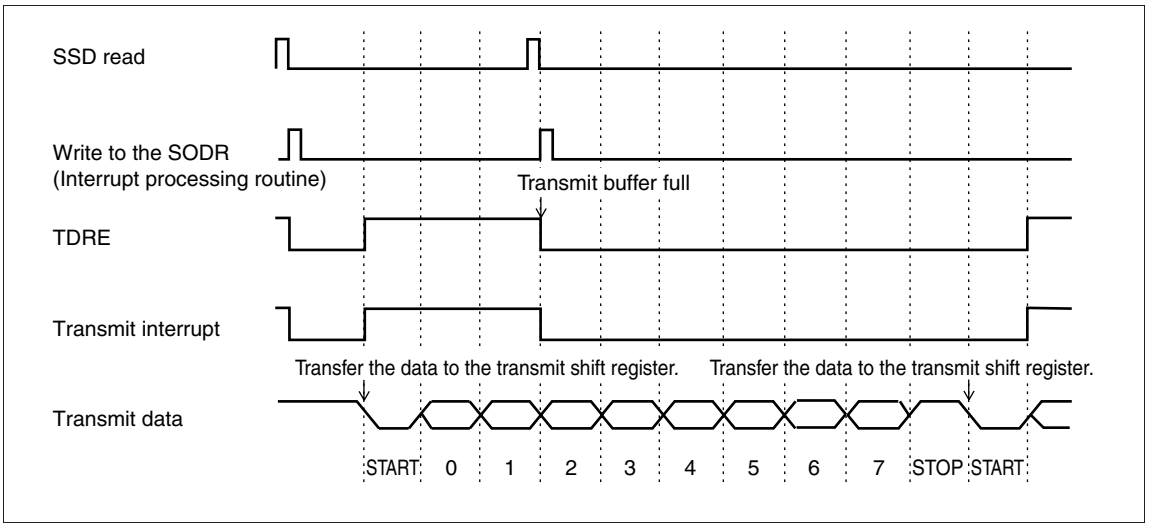
Figure 10.7-1 Operation of operation mode 0, 1, 3



■ Transmit operation

Writing transmit data to the SODR register after reading from SSD register transfers the data written in the SODR to the transmit shift register and initiates a parallel-serial conversion process. The converted transmit data is sent to the serial data output pin with its LSB (Least Significant Bit) followed by other bits (LSB first). When the SODR register gets ready for the next data, the TDRE bit is set to "1" and an interrupt request is issued to CPU (if interrupt enabled, SSD: TIE = "1"). Figure 10.7-2 "Transmit operation in mode 0, 1, 3 " shows the transmit operation when mode 1, non-parity and 1 stop bit are selected.

Figure 10.7-2 Transmit operation in mode 0, 1, 3



■ Receive operation

If receive data is received from the serial data input pin, a serial-parallel conversion process is initiated in the internal receive shift register. After the data is normally (stop bit is detected), the receive data is transferred from the internal shift register to the SIDR register and the RDRF bit is set to "1".

If an overrun or a framing error occurs, the receive data is not stored to the SIDR and the ORFE bit is set to "1".

The RDRF and ORFE bits are set when the last stop bit is detected after the completion of the receive operation. If the receive interrupt is enabled (SSD: RIE = "1"), an interrupt request (IRQ 4) is issued to the CPU. If the RDRF bit is set, the receive data has already been stored to the SIDR register.

Figure 10.7-3 "Receive operation in mode 0, 1, 3", Figure 10.7-4 "Operation at overrun error in mode 0, 1, 3" and Figure 10.7-5 "Operation at framing error in mode 0, 1, 3" show receive operations non-parity and 1 stop bit.

Figure 10.7-3 Receive operation in mode 0, 1, 3

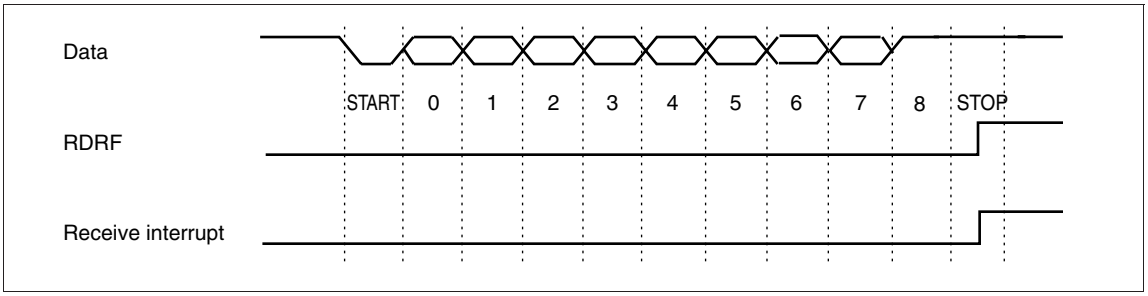


Figure 10.7-4 Operation at overrun error in mode 0, 1, 3

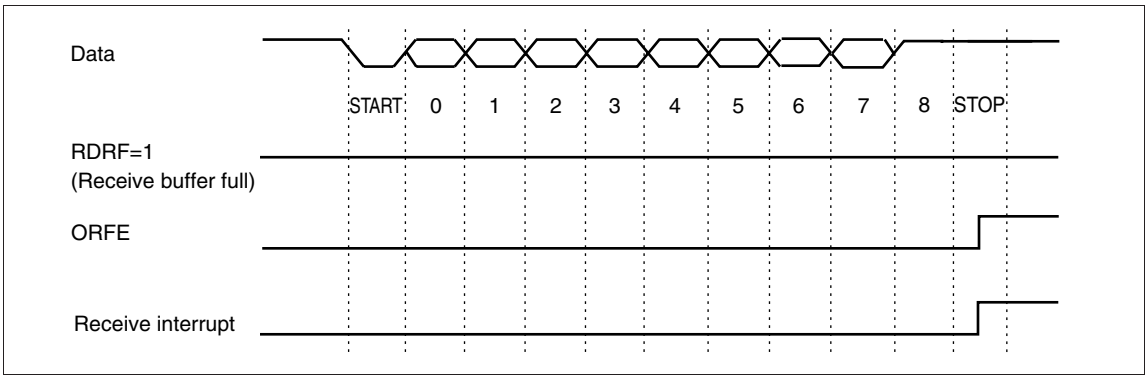
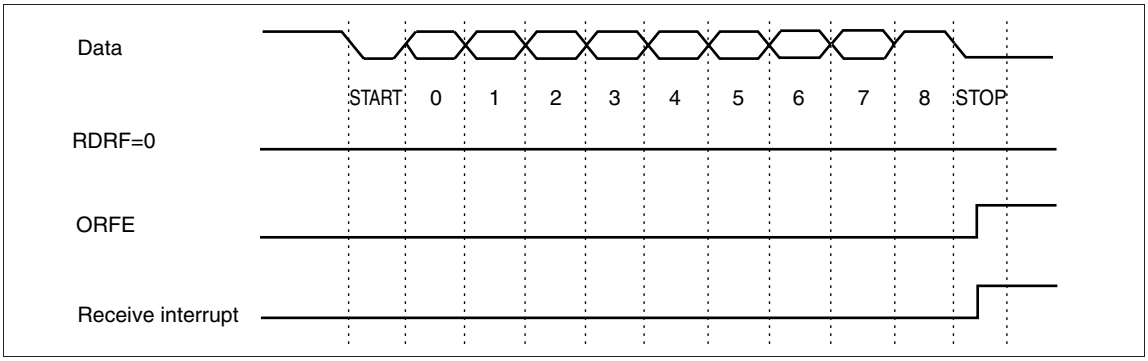


Figure 10.7-5 Operation at framing error in mode 0, 1, 3



Reference:

When the system wakes up from the initialize process caused by reset, an initializing period of 11 shift clocks is needed for initializing the internal control blocks.

10.8 Program Example for UART

This section gives program example for UART.

■ Program example for UART

● Processing description

- Perform serial transmit/receive operation using communication functions of the UART.
- P45/SCK, P44/SO and P43/SI pins are used for communication.
- Set a transmission speed of 150 baud by the internal baud rate generator.
- A character "13_H" is transmitted from the SO pin and triggers the operation by interrupt.
- The baud rate is set with the main clock oscillation frequency (F_{CH}) of 5 MHz

● Coding example

```
PDR4    EQU    000EH           ; Address of the port data register
DDR4    EQU    000FH           ; Address of the port direction register
SMC1    EQU    0020H           ; Address of the serial mode control register 1
SRC     EQU    0021H           ; Address of the serial rate control register
SSD     EQU    0022H           ; Address of the serial status and data register
SIDR    EQU    0023H           ; Address of the serial input data register
SODR    EQU    0023H           ; Address of the serial output data register
SMC2    EQU    0024H           ; Address of the serial mode control register 2
PSEN    EQU    SMC2:5          ; Define the baud rate generator operation
                               ; start/stop bit
ILR2    EQU    007DH           ; Address of the interrupt level setting register
INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF2H
IRQ4    DW     WARI            ; Set interrupt vector.
INT_V   ENDS
;----------Main program-------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
         :
        CLRI                   ; Disable interrupts.
        MOV    ILR2,#11111101B ; Set interrupt level (level 1).
        MOV    SMC1,#01011011B ; Non-parity, 1 stop bit, operating mode 1,
                                 asynchronous, clock output enabled, serial data
                                 output enabled.
        MOV    SRC,#00010100B  ; Proprietary baud rate generator selected.
                                 Set the baud rate at 150 baud.
        MOV    SSD,#00101000B  ; Disable transmit interrupt request, enable
                                 receive interrupt request.
        MOV    SMC2,#00000011B ; Stop UART operation, select UART function and
                                 select the input clock divider of 1/65.
        MOV    SODR,#13H       ; Write transmit data (13H).
        SETB   PSEN            ; Start UART operation.
        SETI                   ; Enable interrupts.
         :
;----------Interrupt processing routine---------------------------------------
WARI    PUSHW  A               ; Save A and T.
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A               ; Restore A and T.
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

CHAPTER 11

EXTERNAL INTERRUPT CIRCUIT (EDGE)

This chapter describes the functions and operation of the external interrupt circuit.

11.1 "Overview of the External Interrupt Circuit"

11.2 "Block Diagram of the External Interrupt Circuit"

11.3 "Structure of the External Interrupt Circuit"

11.4 "External Interrupt Circuit Interrupts"

11.5 "Operation of the External Interrupt Circuit"

11.6 "Program Example for the External Interrupt Circuit"

11.1 Overview of the External Interrupt Circuit

The external interrupt circuit detects edges on the signals input to the two external interrupt pins and generates the corresponding interrupt requests to the CPU.

■ Functions of the external interrupt circuit

The function of the external interrupt circuit is to detect specified edges on signals input to the external interrupt pins and to generate interrupt requests to the CPU. These interrupts can cancel standby mode and return the device to the normal operating state (RUN state).

External interrupt pins: 2 pins (P42/PWC/INT1 and P46/INT0)

External interrupt sources: Input of a specified edge (rising edge or falling edge) on the signal input to an external interrupt pin.

Interrupt control: Output of external interrupt requests is enabled or disabled by the interrupt request enable bits in external interrupt control register (EIC).

Interrupt flags: Detection of specified edges sets the external interrupt request flag bits in external interrupt control register (EIC).

Interrupt request: Separate interrupt request is generated for each external interrupt source (IRQ0, IRQ1).

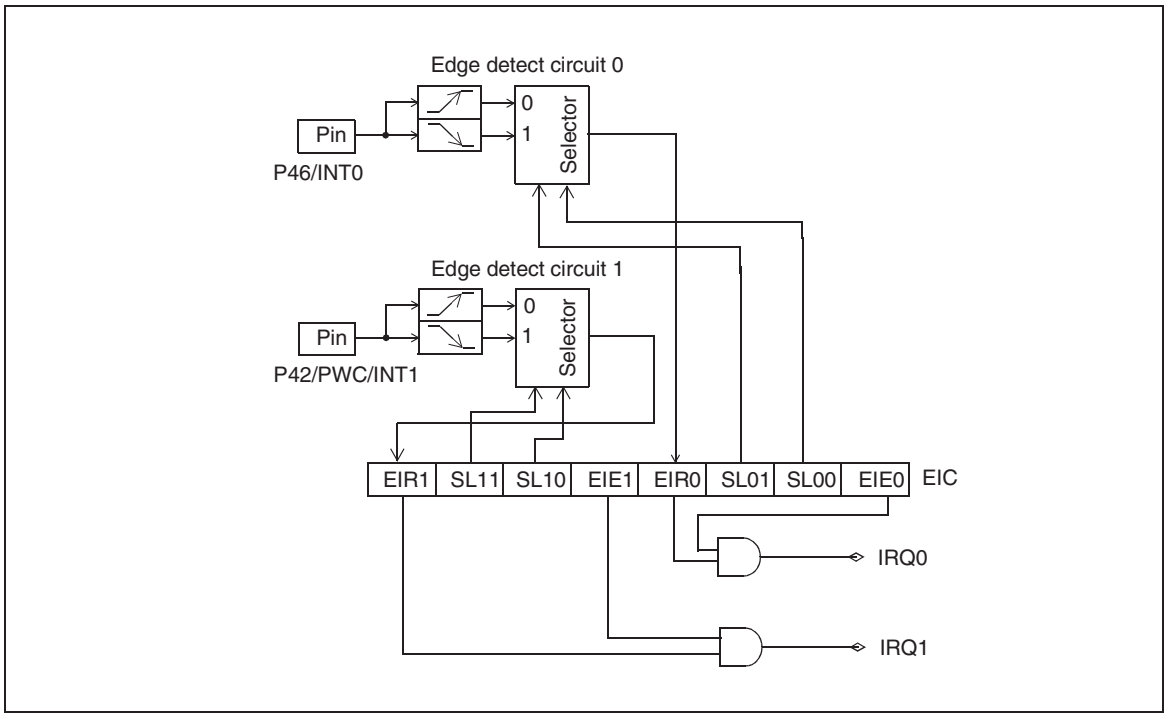
11.2 Block Diagram of the External Interrupt Circuit

The external interrupt circuit consists of the following two elements:

- **Edge detect circuit 0, 1**
- **External interrupt control register (EIC)**

■ Block diagram of the external interrupt circuit

Figure 11.2-1 Block diagram of the external interrupt circuit



● Edge detect circuit

If the polarity of an edge on the input signal to one of the external interrupt pins (INT0 - INT1) matches the edge polarity specified for the pin in the EIC register (SL01, SL00, SL11, SL10), the edge detect circuit sets the corresponding external interrupt request flag bit (EIR0 - EIR1) to "1".

● EIC register

The EIC register is used for operations such as edge selection, enabling or disabling interrupt requests, and checking interrupt requests.

11.3 Structure of the External Interrupt Circuit

This section describes the pins, pin block diagram, register, and interrupt sources of the external interrupt circuit.

■ External interrupt circuit pins

The external interrupt circuit has two external interrupt pins.

The external interrupt pins can function either as external interrupt inputs (hysteresis inputs) or general I/O ports.

Although the P42/PWC/INT1 and P46/INT0 pins continuously function as external interrupt inputs, the external interrupt circuit does not output interrupts if output of interrupt requests is disabled for the pin. The pin states can be read directly from the port data register (PDR4) at any time.

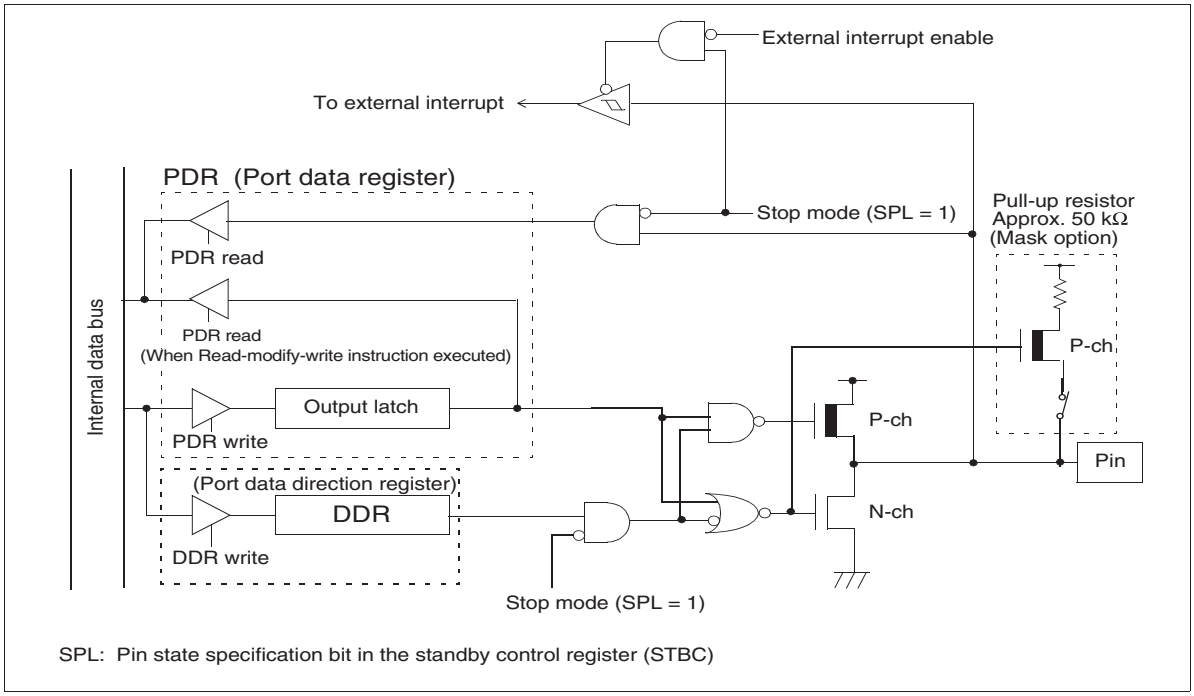
Table 11.3-1 External interrupt circuit pins

External interrupt pin	When used as an external interrupt input (interrupt requests enabled)	When used as general I/O port (interrupt requests disabled)
P46/INT0	INT0 (EIC: EIE0 = "1")	P46 (EIC: EIE0 = "0")
P42/PWC/INT1	INT1 (EIC: EIE1 = "1")	P42 (EIC: EIE1 = "0")

INT0 - INT1: The external interrupt circuit generates the interrupt request when an edge of the specified polarity is detected on the pin.

■ Block diagram of the external interrupt circuit pins

Figure 11.3-1 Block diagram of the external interrupt circuit pins



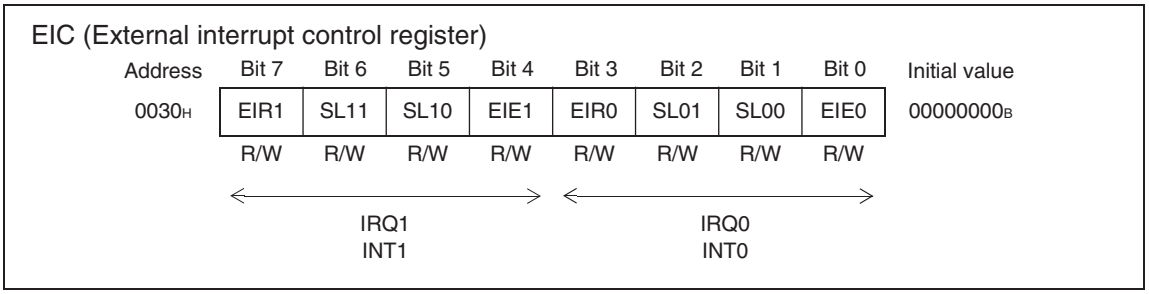
SPL: Pin state specification bit in the standby control register (STBC)

Reference:

Pins with a pull-up resistor go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".

■ External interrupt circuit register

Figure 11.3-2 External interrupt circuit register



■ External interrupt circuit interrupt sources

IRQ0:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT0 when output of interrupt requests is enabled.

IRQ1:

This interrupt request is generated if an edge of the selected polarity is input to the external interrupt pin INT1 when output of interrupt requests is enabled.

11.3.1 External Interrupt Control Register (EIC)

External interrupt control register (EIC) is used to select the edge polarity and to control interrupts for external interrupt pins (INT0, INT1).

■ External interrupt control register (EIC)

Figure 11.3-3 External interrupt control register (EIC)

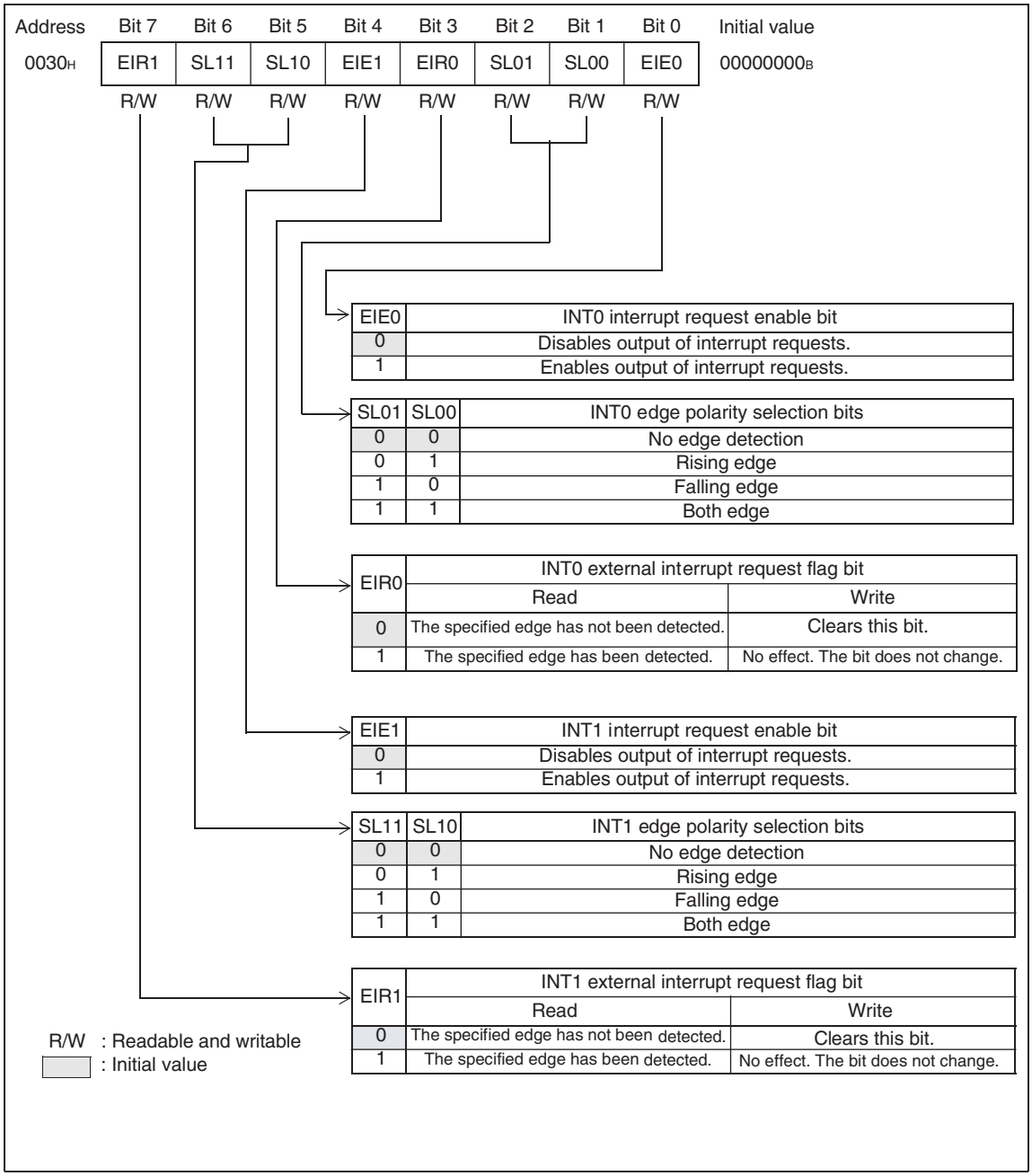


Table 11.3-2 External interrupt control register (EIC) bits

Bit		Function
Bit 7	EIR1: INT1 external interrupt request flag bit	• This bit is set to "1" when the edge selected by INT1 edge polarity selection bits (SL11, SL10) is input to external interrupt pin INT1. • An interrupt request is output when both this bit and INT1 interrupt request enable bit (EIE1) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
Bit 6 Bit 5	SL11, SL10: INT1 edge polarity mode selection bits	• Controls the mode of the input edge polarity of INT1 pin. • Writing "00_B" selects no edge detection, "01_B" selects rising edge mode, "10_B" selects falling edge mode or "11_B" selects both edge mode. • Always write "0" into EIR1 when changing these bits.
Bit 4	EIE1: INT1 interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is generated when both this bit and INT1 external interrupt request flag bit (EIR1) are "1".
Bit 3	EIR0: INT0 external interrupt request flag bit	• This bit is set to "1" when the edge selected by INT0 edge polarity selection bits (SL01, SL00) is input to external interrupt pin INT0. • An interrupt request is output when both this bit and INT10 interrupt request enable bit (EIE0) are "1". • Writing "0" clears the bit. Writing "1" has no effect and does not change the bit value.
Bit 2 Bit 1	SL01, SL00: INT0 edge polarity mode selection bits	• Controls the mode of the input edge polarity of INT0 pin. • Writing "00_B" selects no edge detection, "01_B" selects rising edge mode, "10_B" selects falling edge mode or "11_B" selects both edge mode. • Always write "0" into EIR0 when changing these bits.
Bit 0	EIE0: INT0 interrupt request enable bit	• Enables or disables output of interrupt requests to the CPU. An interrupt request is generated when both this bit and INT0 external interrupt request flag bit (EIR0) are "1".

11.4 External Interrupt Circuit Interrupts

The external interrupt circuit can generate interrupt requests when it detects a specified edge on the signal input to an external interrupt pin.

■ Interrupts when the external interrupt circuit is operating

On detecting a specified edge on an external interrupt input, the external interrupt circuit sets the corresponding external interrupt request flag bit (EIC: EIR0 - EIR1) to "1". An interrupt request to the CPU (IRQ0 - IRQ1) is generated at this time if the corresponding interrupt request enable bit is enabled (EIC: EIE0 - EIE1 = "1"). Always write "0" to the corresponding external interrupt request flag bit in the interrupt processing routine to clear the interrupt request.

Note:

When enabling interrupts (EIE0 - EIE1 = "1") after exit of a reset, always clear the corresponding external interrupt request flag bit (EIR0 - EIR1 = "0") at the same time.

Interrupt processing cannot return if the external interrupt request flag bit is "1" and the interrupt request enable bit is enabled. Always clear the external interrupt request flag bit.

Reference:

Cancelling stop mode using an interrupt is only possible using the external interrupt circuit.

- An interrupt request is generated immediately if the external interrupt request flag bit is "1" when the interrupt request enable bit is changed from disabled to enabled ("0" --> "1").

■ Register and vector table for the external interrupt circuit interrupts

Table 11.4-1 Register and vector table for the external interrupt circuit interrupts

Interrupt	Interrupt level setting register			Vector table address	
	Register	Setting Bits		Upper	Lower
IRQ0	ILR1 ($007C_H$)	L01 (Bit 1)	L00 (Bit 0)	$FFFA_H$	$FFFB_H$
IRQ1		L11 (Bit 3)	L10 (Bit 2)	$FFF8_H$	$FFF9_H$

See Section 3.4.2 "Interrupt Processing" for details on the operation of interrupts.

■ Notes when changing edge polarity selection

When changing the edge polarity for INT0 to INT1, always write "0" into the corresponding EIR bits. This will prevent from accidentally creating an interrupt.

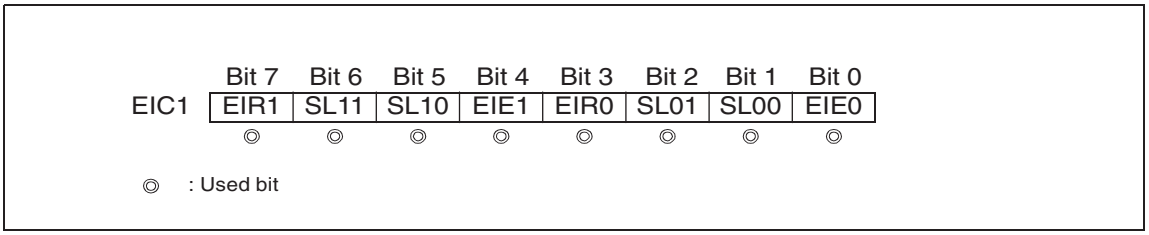
11.5 Operation of the External Interrupt Circuit

The external interrupt circuit can detect a specified edge on a signal input to an external interrupt pin.

■ Operation of the external interrupt circuit

Figure 11.5-1 "External interrupt circuit settings" shows the settings required to operate the external interrupt circuit.

Figure 11.5-1 External interrupt circuit settings

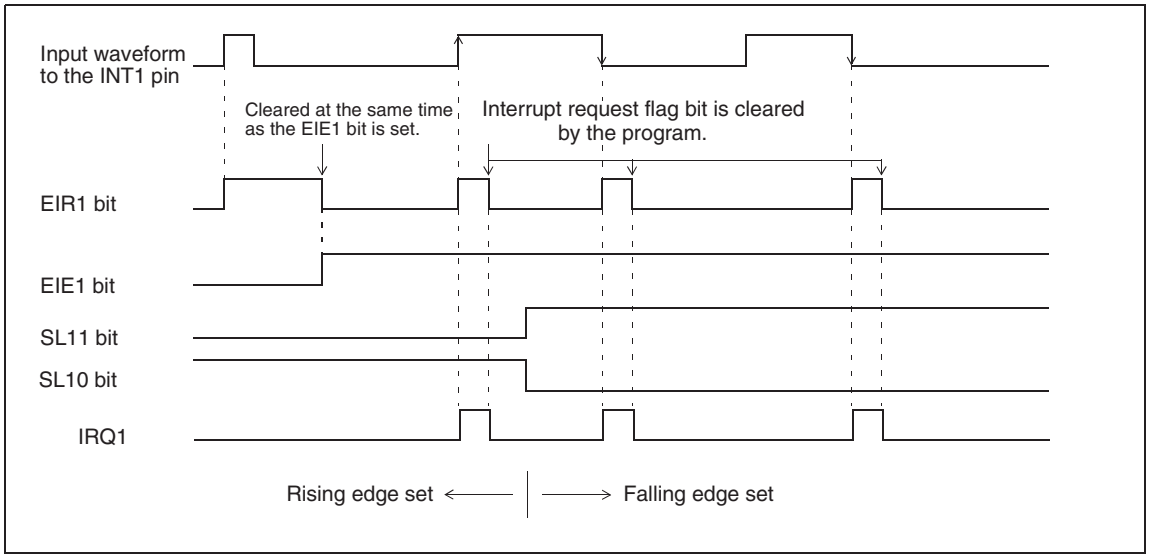


If the polarity of an edge on the input signal to one of the external interrupt pins (INT0 - INT1) matches the edge polarity specified for the pin in the external interrupt control register (EIC: SL11, SL10, SL01, SL00), the external interrupt circuit sets the external interrupt request flag bit (EIC: EIR0 - EIR1) to "1".

The external interrupt request flag bit is set when the edge polarity match occurs, regardless of the value of the interrupt request enable bit (EIC: EIE0 - EIE1).

Figure 11.5-2 "External interrupt (INT1) operation" shows the operation when an external interrupt is input to the INT1 pin.

Figure 11.5-2 External interrupt (INT1) operation



Reference:

The pin state can be read directly from the port data register (PDR4), even when used as an external interrupt input.

11.6 Program Example for the External Interrupt Circuit

This section gives a program example for the external interrupt circuit.

■ Program example for the external interrupt circuit

● Processing description

- Generates interrupts on detecting a rising edge on pulses input to the INT1 pin.

● Coding example

```
EIC1    EQU    0030H           ; External interrupt control register 1

EIR1    EQU    EIC1:7          ; Defines the external interrupt request flag bit.
SL10    EQU    EIC1:5          ; Defines the edge polarity selection bit.
EIE1    EQU    EIC1:4          ; Defines the interrupt request enable bit.

ILR1    EQU    007CH           ; Set interrupt level setting register 1.

INT_V   DSEG   ABS             ; [DATA SEGMENT]
        ORG    0FFF8H
IRQ1    DW     WARI            ; Set INT1 interrupt vector.
INT_V   ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
                               ; Stack pointer (SP) etc. are already initialized.
         :
        CLRI                   ; Disable interrupts.
        CLRB   EIR1            ; Clear interrupt request flag.
        MOV    ILR1,#11110111B ; Set interrupt level (level 1).
        SETB   SL10            ; Select rising edge.
        SETB   EIE1            ; Enable output of interrupt requests.
        SETI                   ; Enable interrupts.
         :
;-----Interrupt processing routine---------------------------------------------
WARI    CLRB   EIE1            ; Clear INT1 interrupt request flag.
        PUSHW  A
        XCHW   A,T
        PUSHW  A
         :
        User processing
         :
        POPW   A
        XCHW   A,T
        POPW   A
        RETI
        ENDS
;------------------------------------------------------------------------------
        END
```

CHAPTER 12

LCD CONTROLLER/DRIVER

This chapter describes the functions and operation of the LCD controller/driver.

12.1 "Overview of LCD Controller/Driver"
12.2 "Block Diagram of LCD Controller/Driver"
12.3 "Structure of LCD Controller/Driver"
12.4 "Operation of LCD Controller/Driver"
12.5 "Program Example for LCD Controller/Driver"

12.1 Overview of LCD Controller/Driver

The LCD controller/driver includes 21 bytes of on-chip display data in memory, the contents of which control an LCD via 42 segment and 4 common outputs. The function can drive an LCD panel directly, using one of three selectable duty ratios.

■ LCD controller/driver function

The LCD controller/driver function displays the contents of a display data memory directly to the LCD (Liquid Crystal Display) panel by segment and common outputs.

- LCD can be driven directly.
- Built-in voltage divider for LCD driving voltage. Can be connected to the external voltage divider.
- Up to 42 segment outputs (SEG0 to SEG41) and four common outputs (COM0 to COM3) may be used.
- Built-in display RAM: 21 bytes (42 x 4 bits)
- Three selectable duty ratios (1/2, 1/3, and 1/4). Not all duty ratios are available with all bias settings.
- SEG20 to SEG41 can be used as general-purpose port (option).

Table 12.1-1 "Bias and duty ratio combinations" shows the duty ratios available with various bias settings.

Table 12.1-1 Bias and duty ratio combinations

Bias	Duty ratio		
	1/2 duty ratio	1/3 duty ratio	1/4 duty ratio
1/2 bias	○	X	X
1/3 bias	X	○	○

○: Recommended mode
X: Do not use

Note:

P00/SEG20 to P07/SEG27, P10/SEG28 to P17/SEG35 and P20/SEG36 to P25/SEG41 are set as N-ch open-drain I/O by mask option, they cannot be used as LCD segment output.

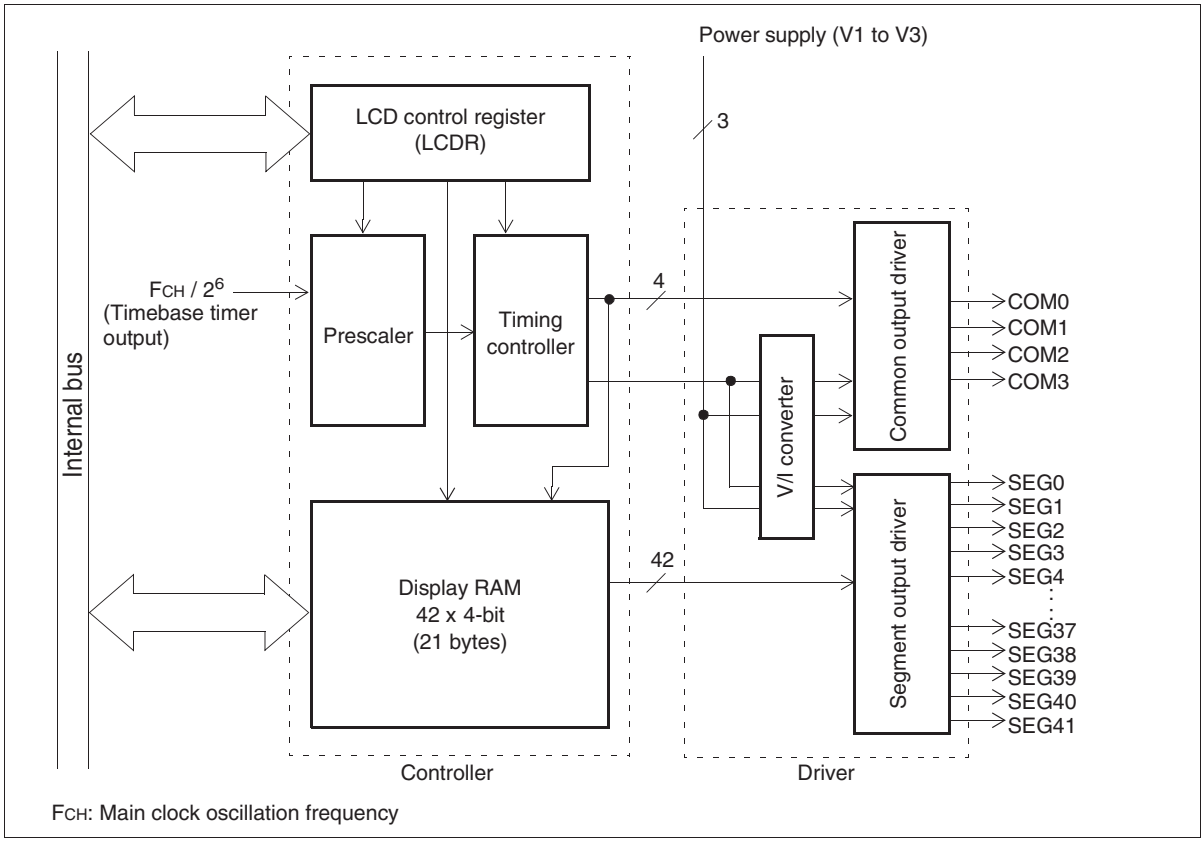
12.2 Block Diagram of LCD Controller/Driver

The LCD controller/driver is made up of seven blocks listed below. Functionally, the circuit can be broken into two major sections: the controller section, which generates LCD segment and common signals based on the current contents of display RAM, and the driver section, which develops sufficient drive to operate the display.

- **LCD control register (LCDR)**
- **Display RAM**
- **Prescaler**
- **Timing controller**
- **V/I converter**
- **Common output driver**
- **Segment output driver**

■ Block diagram of LCD controller/driver

Figure 12.2-1 Block diagram of LCD controller/driver



● LCD control register (LCDR)

This register is used to control the LCD drive supply voltage, select display blanking/non-blanking, select the display mode, and select the LCD clock cycle.

- Display RAM

 This 42 x 4-bit block of RAM controls the segment output signals. Its contents are automatically read out to the segment outputs in synchronous with the timing of the selected common signal.

- Prescaler

 The prescaler generates one of the 4 frame frequencies according to the LCD control register setting.

- Timing controller

 This block controls the segment and common signals based on the frame frequency and LCD control register settings.

- V/I converter

 This circuit generates alternating current waveforms from the voltage signals it receives from the timing controller to drive the LCD.

- Common output driver

 This block contains the drivers for the LCD common pins.

- Segment output driver

 This block contains the drivers for the LCD segment pins.

- Voltage divider (optional)

 This voltage divider is used to provide the divided LCD driving voltage. The voltage divider can be connected externally.

12.2.1 LCD Controller/Driver Internal Voltage Divider

LCD driver supply voltage can be taken from an internal voltage divider (external voltage divider may also be used).

■ Internal voltage divider

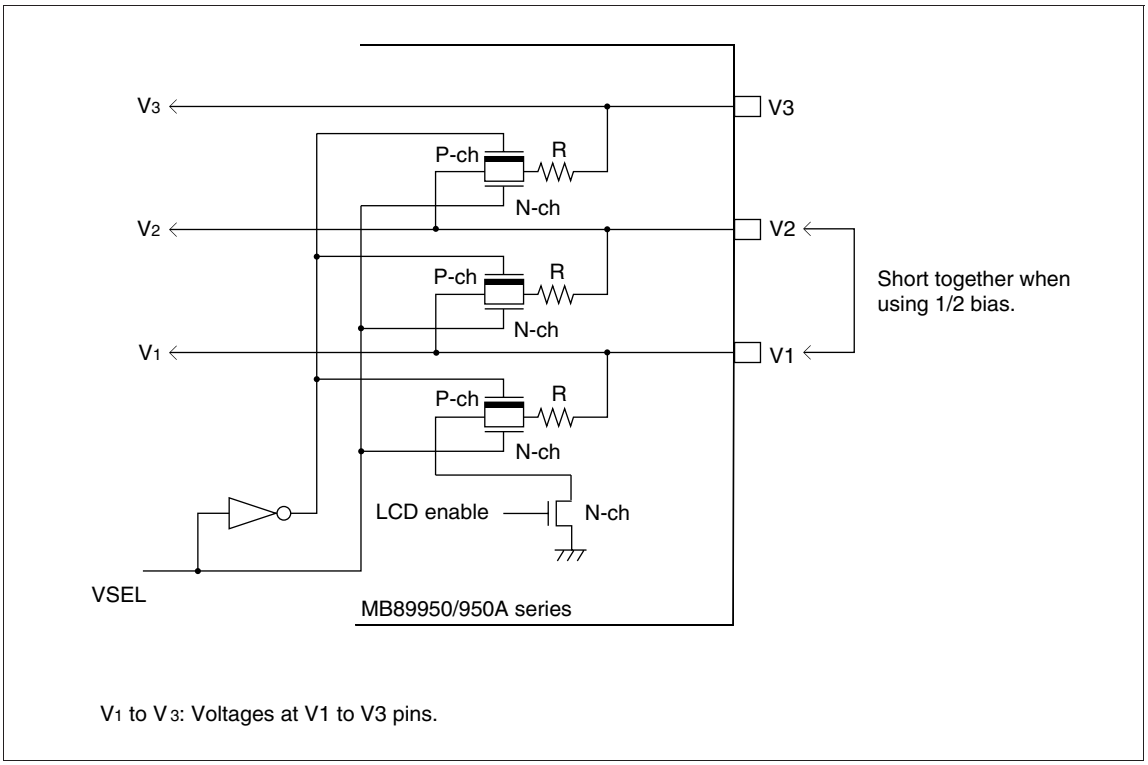
In these devices, external voltage divider may also be connected at pins V1 through V3.

The selection of internal or external voltage divider is made by the drive supply voltage control bit of LCD control register (LCDR: VSEL). VSEL = "1" connects the internal voltage divider. Set VSEL to "1" when you want to use the internal voltage divider only (when no external voltage divider is connected).

The LCD enable is inactive when LCD operation is stopped (LCDR: MS1, MS0 = "00_B"), or in stop mode (STBC: STP = "1"). Pin V2 and V1 should be shorted together when using the 1/2 bias setting.

Figure 12.2-2 "Internal voltage divider equivalent circuit" shows an equivalent circuit of the internal voltage divider.

Figure 12.2-2 Internal voltage divider equivalent circuit

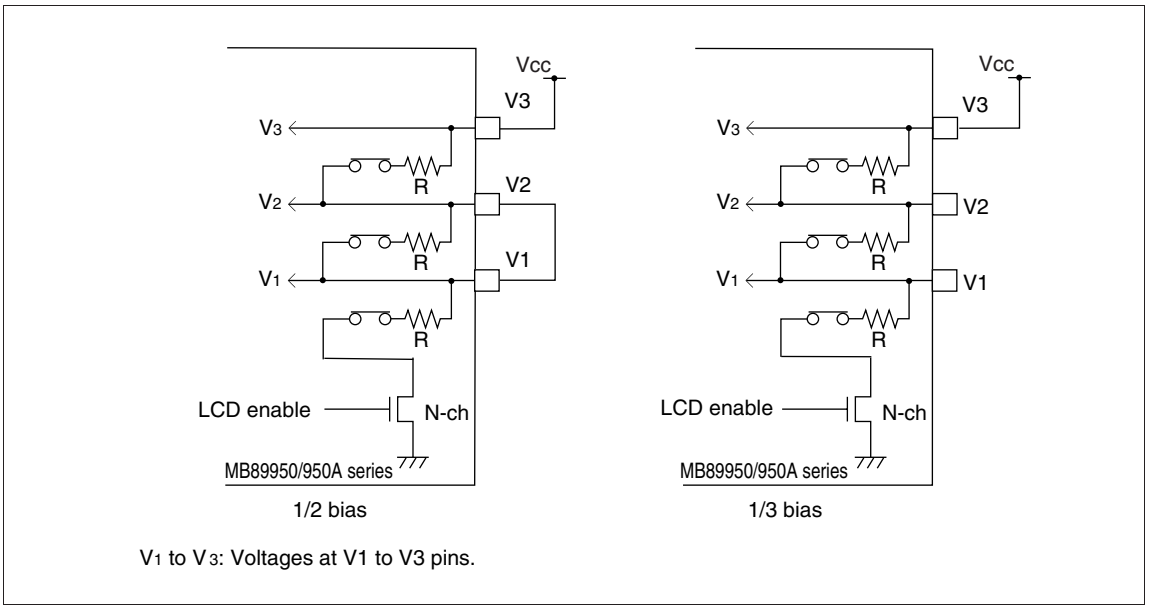


V1 to V3: Voltages at V1 to V3 pins.

■ Use of internal voltage divider

Figure 12.2-3 "Use of internal voltage divider" shows the voltage divider circuits for 1/2 and 1/3 bias. As shown in this figure, in the 1/2 bias mode (with LCD enabled) V2 and V1 will be 1/2 of V3 (V3 is the LCD operating voltage, which is V_{CC} in this configuration). In the 1/3 bias mode, V1 is 1/3 of V3, and V2 is 2/3 of V3.

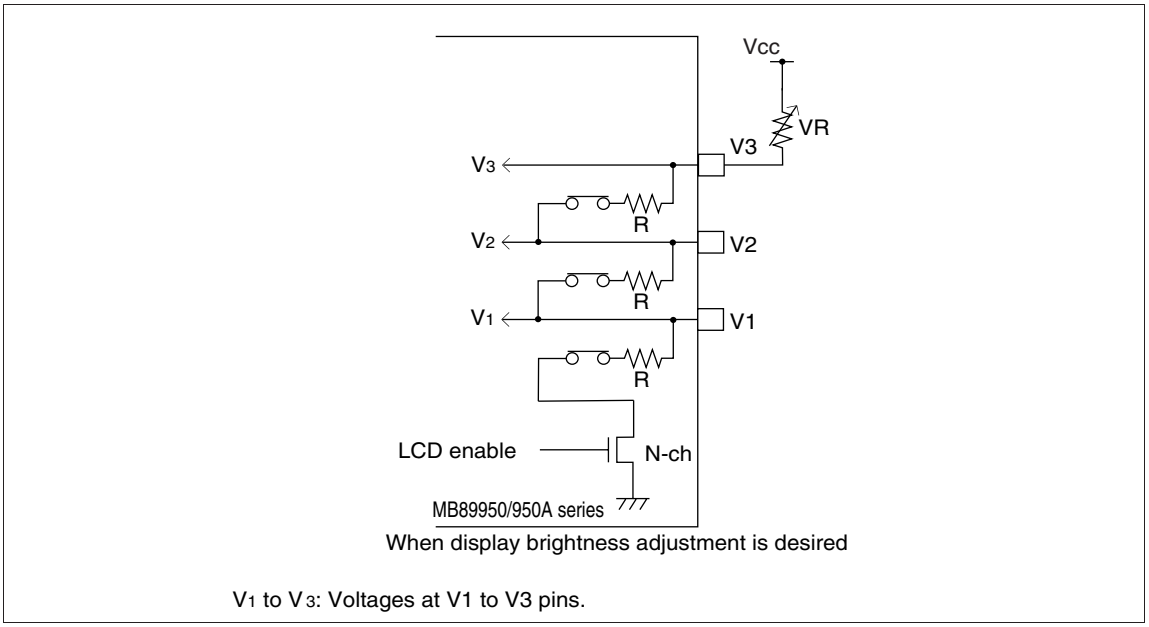
Figure 12.2-3 Use of internal voltage divider



■ Display brightness adjustment when internal voltage divider is used

When internal voltage divider does not provide sufficient LCD display brightness, connect an external brightness adjust variable resistor between V_{CC} and V3 as shown in Figure 12.2-4 "Use of internal voltage divider with brightness adjustment".

Figure 12.2-4 Use of internal voltage divider with brightness adjustment



12.2.2 LCD Controller/Driver External Voltage Divider

External voltage divider can also be used with devices that have internal voltage divider. Display brightness can be adjusted by a variable resistor(VR) connected between the V_{CC} and V3 pins.

■ External voltage divider

When you do not wish to use the internal voltage divider, external voltage divider resistors can be connected at the LCD drive voltage supply pins (V1 to V3). Figure 12.2-5 "External voltage divider connection" shows connection for external voltage divider for the two biasing modes, and Table 12.2-1 "LCD drive voltages and biasing modes" lists the corresponding LCD drive voltages.

Figure 12.2-5 External voltage divider connection

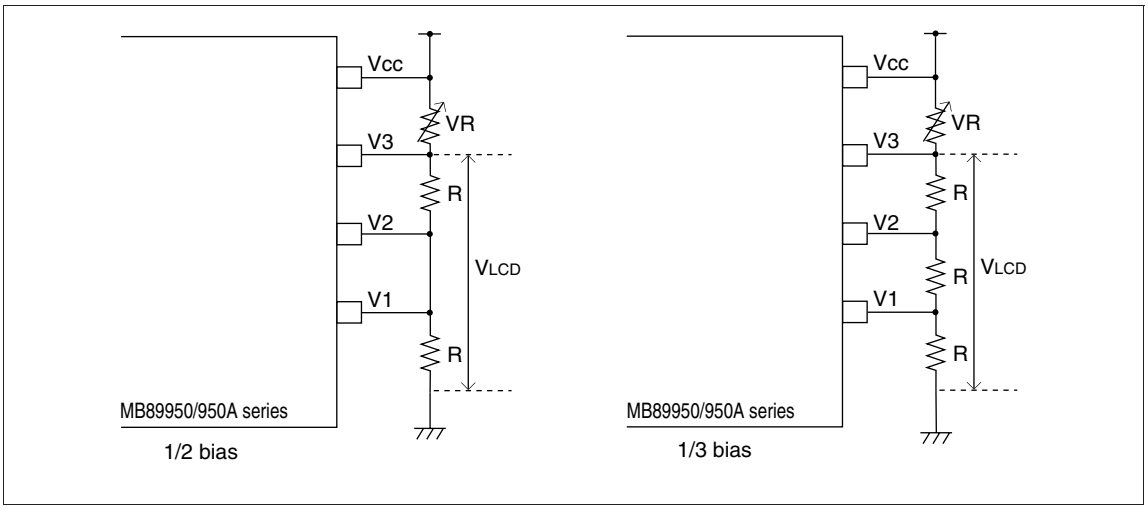


Table 12.2-1 LCD drive voltages and biasing modes

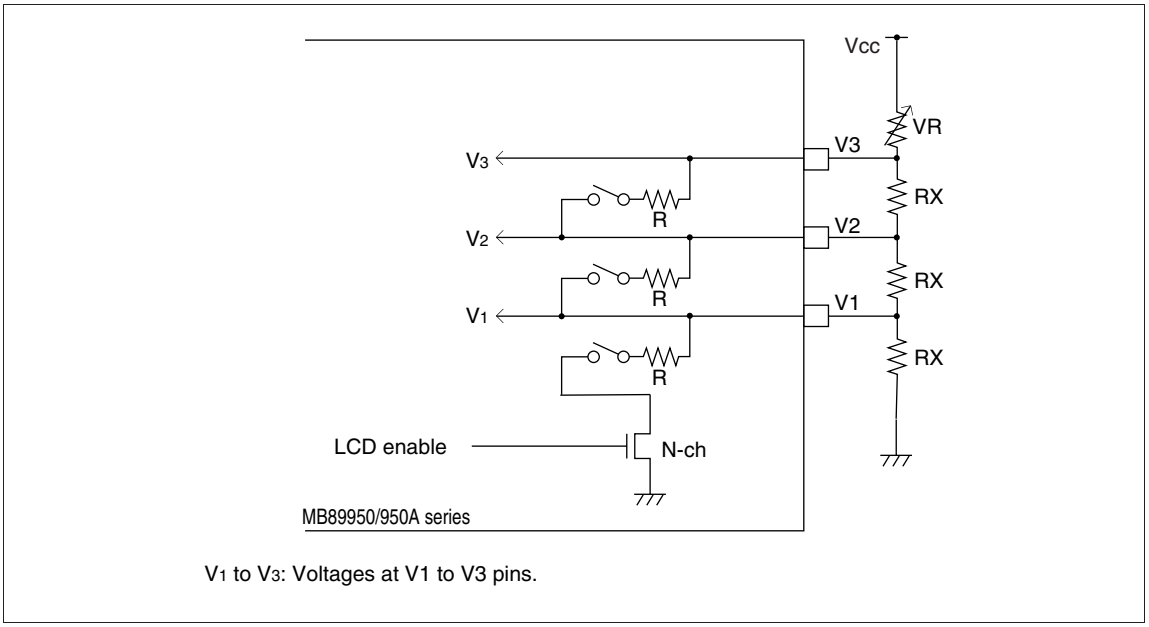
Bias	LCD drive voltages		
	V3	V2	V1
1/2 bias	VLCD	1/2VLCD	1/2VLCD
1/3 bias	VLCD	2/3VLCD	1/3VLCD

V1 to V3: Voltages at pins V1 to V3.
VLCD: LCD operating voltage

■ Use of external voltage divider

Figure 12.2-6 "External voltage divider connection" shows an external voltage divider connection.

Figure 12.2-6 External voltage divider connection



Note:

To preclude the external voltage divider from being affected by the internal voltage divider, the LCD drive supply voltage control bit of LCD control register (LCDR: VSEL) must be written to "0" to isolate it from the entire internal voltage divider.

Reference:

The resistance of RX in the external voltage divider depends on the LCD used. Select an appropriate value.

12.3 Structure of LCD Controller/Driver

This section describes the pins, pin block diagrams, registers, and display RAM of the LCD controller/driver.

■ LCD controller/driver pins

The LCD controller/driver uses 4 common output pins (COM0 to COM3), 42 segment output pin (SEG0 to SEG41), and 3 LCD driving power supply pins (V1 to V3).

● COM0, COM1, COM2, and COM3 pins

COM0 to COM3 can function LCD common output pins (COM0 to COM3).

● SEG0 to SEG19, P00/SEG20 to P07/SEG27, P10/SEG28 to P17/SEG35 and P20/SEG36 to P25/SEG41

P00/SEG20 to P07/SEG27, P10/SEG28 to P17/SEG35 and P20/SEG36 to P25/SEG41 pins can function either as N-ch open-drain I/O ports (P00 to P07, P10 to P17 and P20 to P25) and LCD segment output pins (SEG20 to SEG41). The selection, however is made as a mask option.

Note:

When these pins are used as LCD segment outputs, the corresponding port data registers (PDR0, PDR1 and PDR2) should be set to all "1" to turn the output transistors "OFF".

● P32/V1, P33/V2 and V3

V1, V2 and V3 pins are the LCD driving power supply pins. The P32/V1 and P33/V2 can function either as N-ch open-drain I/O ports (P32 and P33) and LCD driving power supply pins (V1 and V2). The selection, however is made by setting LCDR: PSEL bit.

■ Block diagrams of LCD controller/driver pins

Figure 12.3-1 Block diagram of LCD controller/driver pins (dedicated common/segment output pins COM0 to COM3 and SEG0 to SEG19)

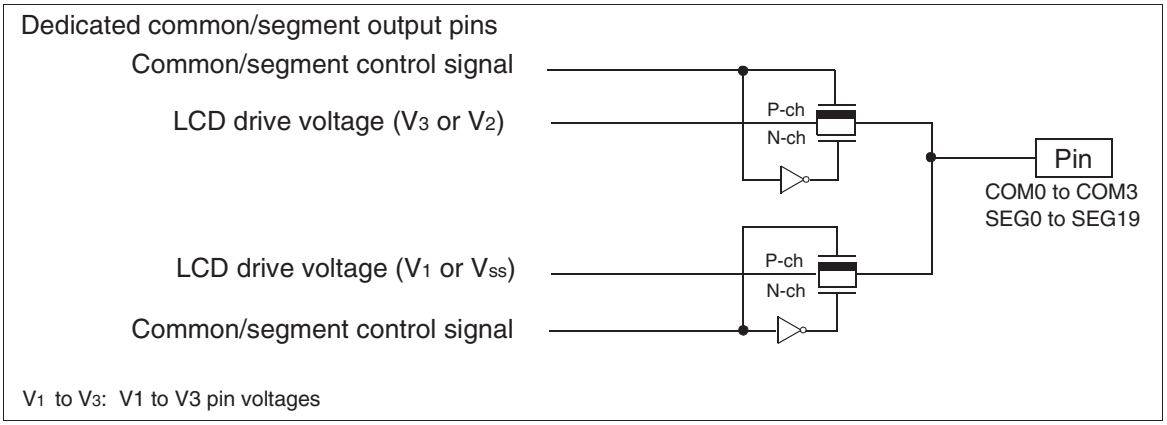


Figure 12.3-2 Block diagram of LCD controller/driver pins (SEG20 to SEG41)

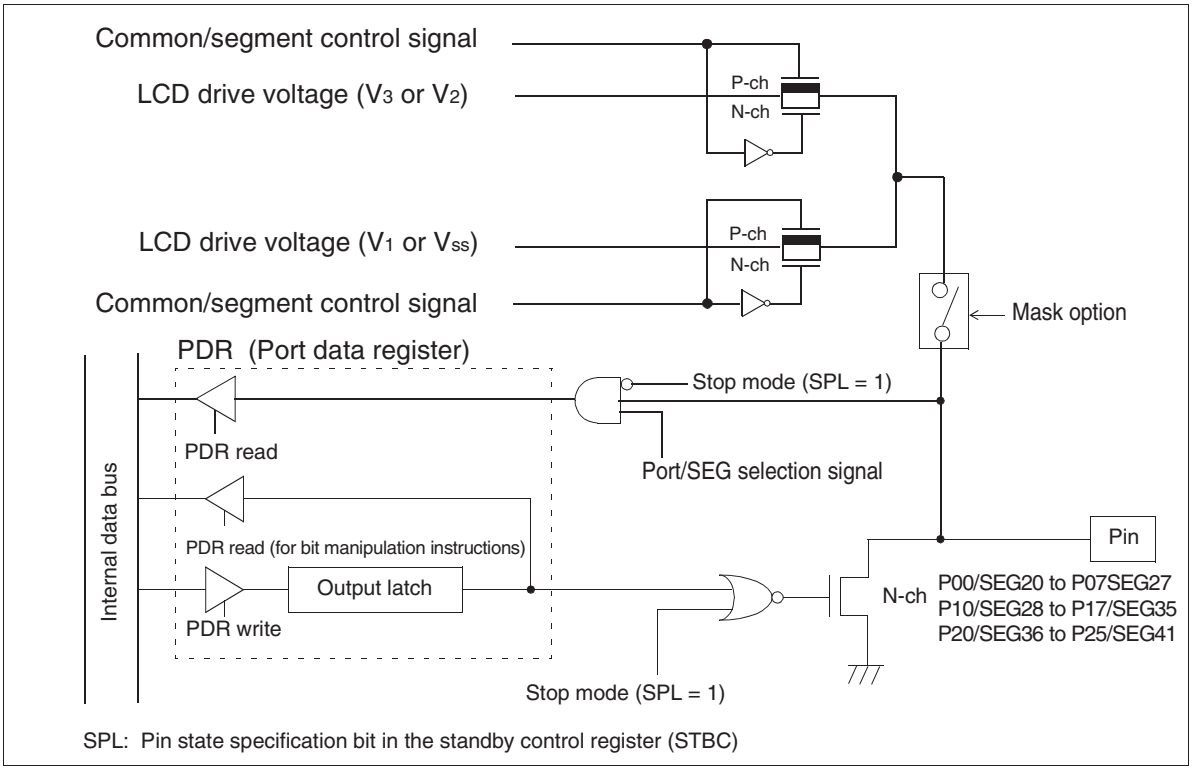
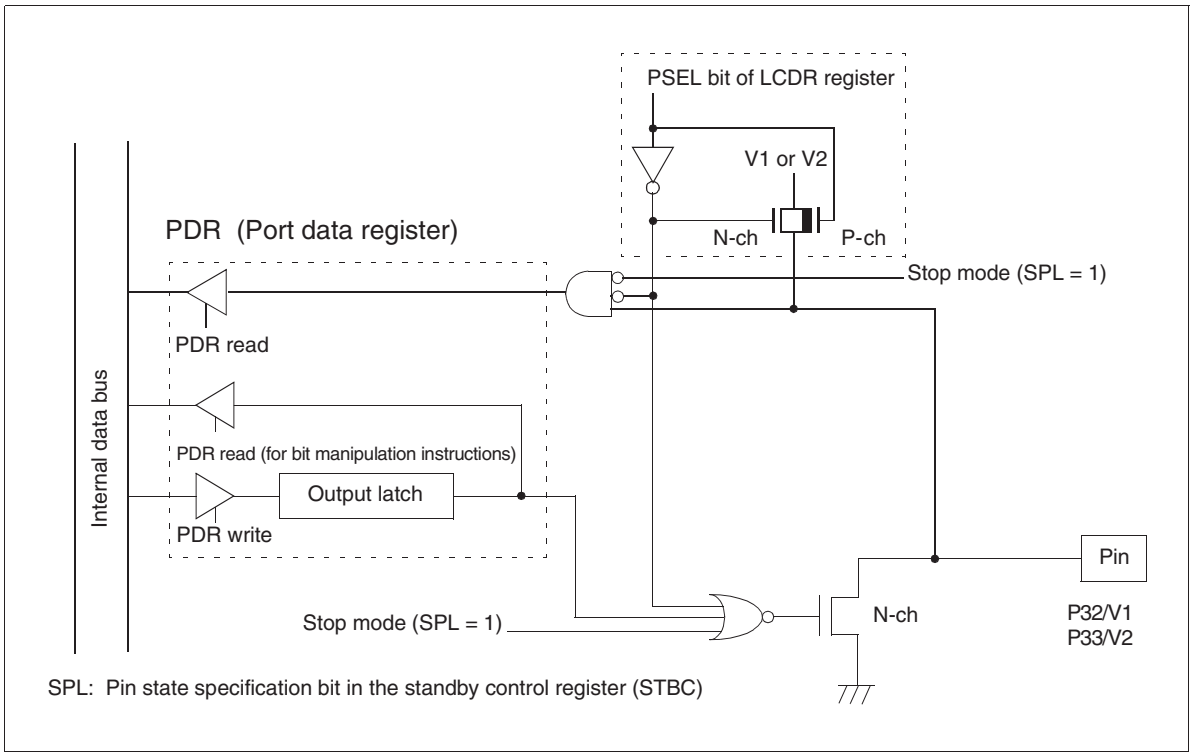


Figure 12.3-3 Block diagram of LCD controller/driver pin (P32/V1 and P33/V2)



■ LCD controller/driver registers

Figure 12.3-4 LCD controller/driver registers

LCDR (LCD control register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0079H	RESV	PSEL	VSEL	BK	MS1	MS0	FP1	FP0	-0010000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SEGR (Segment output select register)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
007AH	—	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG00	-0000000B
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	

R/W : Readable and writable
— : Unused

■ LCD controller/driver RAM

LCD controller/driver has 42 x 4-bit of internal display RAM in which the data used to generate the segment output signals is stored.

12.3.1 LCD Control Register (LCDR)

LCD control register (LCDR) is used to select the frame cycle, control the LCD drive supply voltage, select display blanking/non-blanking, and select the display mode.

■ LCD control register (LCDR)

Figure 12.3-5 LCD control register (LCDR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
0079H	RESV	PSEL	VSEL	BK	MS1	MS0	FP1	FP0	-0010000B
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

FP1	FP0	Frame cycle selection bits	
0	0	$F_{CH}/(2^{11} \times N)$	(610 Hz)
0	1	$F_{CH}/(2^{12} \times N)$	(305 Hz)
1	0	$F_{CH}/(2^{13} \times N)$	(152 Hz)
1	1	$F_{CH}/(2^{14} \times N)$	(76 Hz)

() : Values for F_{CH}=5 MHz, and N=4
N : Number of time divisions
F_{CH} : Main clock frequency oscillation

MS1	MS0	Display mode selection bits
0	0	Stops LCD operation
0	1	1/2 duty ratio output mode (time division N = 2)
1	0	1/3 duty ratio output mode (time division N = 3)
1	1	1/4 duty ratio output mode (time division N = 4)

BK	Display blanking selection bit
0	Displays unblanked
1	Displays blanked

VSEL	Drive supply voltage control bit
0	Uses external voltage divider (internal voltage divider is isolated).
1	Uses internal voltage divider.

PSEL	LCD voltage supply selection bit
0	Selects as LCD power supply pins (V1 and V2)
1	Selects as port pins (P32 and P33)

RESV	Reserved bit
	Always write “0” to this bit.

R/W : Readable and writable
— : Unused
□ : Initial value

Table 12.3-1 LCD control register (LCDR) bit functions

Bit		Function
Bit 7	Reserved bit	• Always write "0" to this bit.
Bit 6	PSEL: LCD power supply selection bit	• Selects P32/V1 and P33/V2 to function either as N-ch open-drain I/O ports (P32, P33) or as LCD power supply pins (V1 and V2).
Bit 5	VSEL: LCD drive supply voltage control bit	• This bit controls the use of the internal voltage divider. Writing a "1" to it enables the use of the internal voltage divider. Writing a "0" to it disables the use of the internal voltage divider. **Note:** This bit must be "0" in order to isolate the internal voltage divider when external voltage divider is used.
Bit 4	BK: Display blanking selection bit	• Blanks/unblanks the LCD. • Setting this bit to "1" (blank) outputs a "deselect" waveform to the LCD segments (which blanks the display).
Bit 3 Bit 2	MS1, MS0: Display mode selection bits	• Selects one of three output waveform duty ratio modes. The mode selected affects the common pins used. Setting both bits to "0" turns off the display (stops LCD controller/driver display operation). **Note:** Before going to a mode in which the selected frame cycle generate clock oscillator is stopped (stop mode, etc.), these bits should be written to "00_B" to turn off the display.
Bit 1 Bit 0	FP1, FP0: Frame cycle selection bits	• These bits select one of four LCD frame cycles. **Note:** To determine this register setting, calculate the optimum frame frequency for the LCD module you are using. Note that the frame cycle is a function of main clock frequency.

12.3.2 Segment Output Select Register (SEGR)

Segment output select register (SEGR) is used to select N-ch open-drain I/O port function or segment output function for P00/SEG20 to P07/SEG27, P10/SEG28 to P17/SEG35 and P20/SEG36 to P25/SEG41, in order to be consistent with mask option.

■ Segment output select register (SEGR)

Figure 12.3-6 Segment output select register (SEGR)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Initial value
007A$_H$	—	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG00	-0000000$_B$
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SEG00	Segment output selection bit
0	Select as N-ch open-drain I/O ports (P00 to P07)
1	Select as segment output (SEG20 to SEG27)

SEG10	Segment output selection bit
0	Select as N-ch open-drain I/O ports (P10 to P13)
1	Select as segment output (SEG28 to SEG31)

SEG11	Segment output selection bit
0	Select as N-ch open-drain I/O ports (P14 to P15)
1	Select as segment output (SEG32 to SEG33)

SEG12	Segment output selection bit
0	Select as N-ch open-drain I/O port (P16)
1	Select as segment output (SEG34)

SEG13	Segment output selection bit
0	Select as N-ch open-drain I/O port (P17)
1	Select as segment output (SEG35)

SEG14	Segment output selection bit
0	Select as N-ch open-drain I/O ports (P20 to P23)
1	Select as segment output (SEG36 to SEG39)

SEG15	Segment output selection bit
0	Select as N-ch open-drain I/O ports (P24 to P25)
1	Select as segment output (SEG40 to SEG41)

R/W : Readable and writable
— : Unused
□ : Initial value

Table 12.3-2 Segment output select register bit functions

Bit		Function
Bit 7	Unused bit	• The read value is indeterminate. • Writing to this bit has no effect on the operation.
Bit 6	SEG15: Segment output selection bit	• Selects P24/SEG40 to P25/SEG41 to function either as N-ch open-drain I/O ports (P24 to P25) or as LCD segment outputs (SEG40 to SEG41). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 5	SEG14: Segment output selection bit	• Selects P20/SEG36 to P23/SEG39 to function either as N-ch open-drain I/O ports (P20 to P23) or as LCD segment outputs (SEG36 to SEG39). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 4	SEG13: Segment output selection bit	• Selects P17/SEG35 to function either as N-ch open-drain I/O port (P17) or as LCD segment output (SEG35). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 3	SEG12: Segment output selection bit	• Selects P16/SEG34 to function either as N-ch open-drain I/O port (P16) or as LCD segment output (SEG34). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 2	SEG11: Segment output selection bit	• Selects P14/SEG32 to P15/SEG33 to function either as N-ch open-drain I/O ports (P14 to P15) or as LCD segment outputs (SEG32 to SEG33). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 1	SEG10: Segment output selection bit	• Selects P10/SEG28 to P13/SEG31 to function either as N-ch open-drain I/O ports (P10 to P13) or as LCD segment outputs (SEG28 to SEG31). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.
Bit 0	SEG00: Segment output selection bit	• Selects P00/SEG20 to P07/SEG27 to function either as N-ch open-drain I/O ports (P00 to P07) or as LCD segment outputs (SEG20 to SEG27). **Note:** The setting of this bit MUST be consistent with mask option. This bit cannot override the mask option.

12.3.3 Display RAM

Display RAM consists of 42 x 4-bit (21 bytes) of display data memory used to generate the segment output signals.

■ Display RAM and output pins

The contents of display RAM are automatically read out and output via the segment outputs in synchronous with the selected common signal timing. A "1" bit is converted to a "select" (display on) voltage and a "0" to a "deselect" (display off) voltage. Since the operation of the LCD is not directly related to the operation of the CPU, display RAM read/write timing can be set by the user. The SEG20 to SEG41 pins that are not made dedicated segment outputs by mask option selection may be used as N-ch open-drain I/O port pins, and the RAM that goes with those pins may be used as regular RAM. (See Table 12.3-3 "Segment outputs, display RAM locations, and sharing port pins".)

Figure 12.3-7 "Segment/common output pins and corresponding display RAM" shows which display RAM bits are associated with each segment and common output pin.

Figure 12.3-7 Segment/common output pins and corresponding display RAM

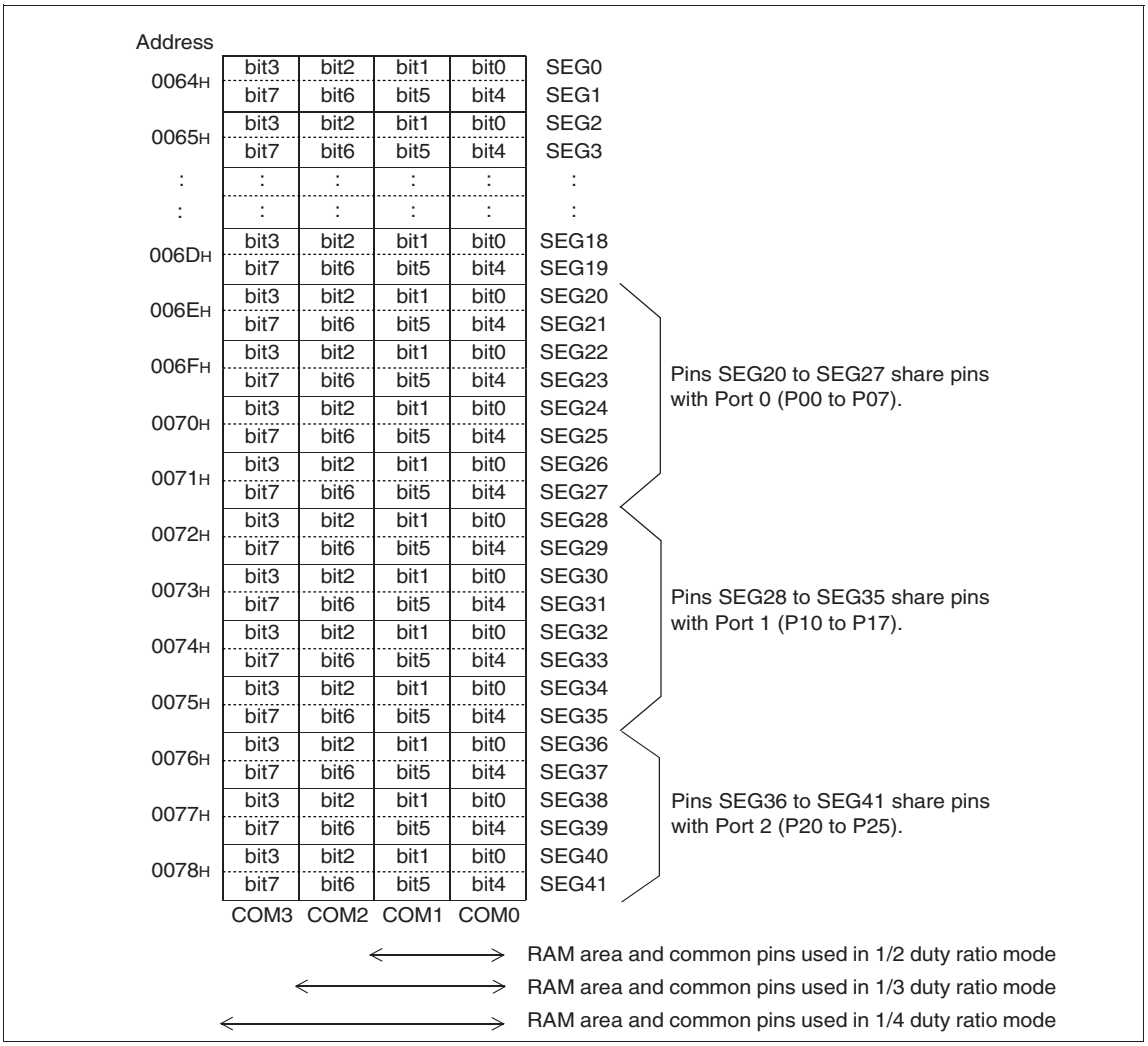


Table 12.3-3 Segment outputs, display RAM locations, and sharing port pins

Segment/common output pins used (mask option)	Corresponding display RAM area	General-purpose ports sharing same pins
SEG0 to SEG19 (20 pins)	64_H to $6D_H$	P00 to P07, P10 to P17, P20 to P25 (22 pins)
SEG0 to SEG19, SEG40 to SEG41 (22 pins)	64_H to $6D_H$ 78_H	P00 to P07, P10 to P17, P20 to P23 (20 pins)
SEG0 to SEG19, SEG36 to SEG41 (26 pins)	64_H to $6D_H$ 76_H to 78_H	P00 to P07, P10 to P17 (16 pins)
SEG0 to SEG27, SEG36 to SEG41 (34 pins)	64_H to 71_H 76_H to 78_H	P10 to P17 (8 pins)
SEG0 to SEG31, SEG36 to SEG41 (38 pins)	64_H to 73_H 76_H to 78_H	P14 to P17 (4 pins)
SEG0 to SEG39 (40 pins)	64_H to 77_H	P24 to P25 (2 pins)
SEG0 to SEG33, SEG36 to SEG41 (40 pins)	64_H to 74_H 76_H to 78_H	P16 to P17 (2 pins)
SEG0 to SEG34, SEG36 to SEG41 (41 pins)	64_H to 78_H	P17 (1 pin)
SEG0 to SEG41 (42 pins)	64_H to 78_H	None

Note:

Locations in the display RAM area that are not required for display data can be used as regular RAM. If any customer wants to choose the mask option combination which is not shown in Table 12.3-3 "Segment outputs, display RAM locations, and sharing port pins", please inform Fujitsu for special testing arrangement.

Table 12.3-4 "Common outputs and display RAM bits used in each duty ratio mode" shows the relationship between duty ratio mode, common outputs, and display RAM.

Table 12.3-4 Common outputs and display RAM bits used in each duty ratio mode

Duty ratio setting	Common outputs used	Display data bit used							
		bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
1/2	COM0 to COM1 (2 pins)	−	−	○	○	−	−	○	○
1/3	COM0 to COM2 (3 pins)	−	○	○	○	−	○	○	○
1/4	COM0 to COM3 (4 pins)	○	○	○	○	○	○	○	○

○ : Used
−: Not used

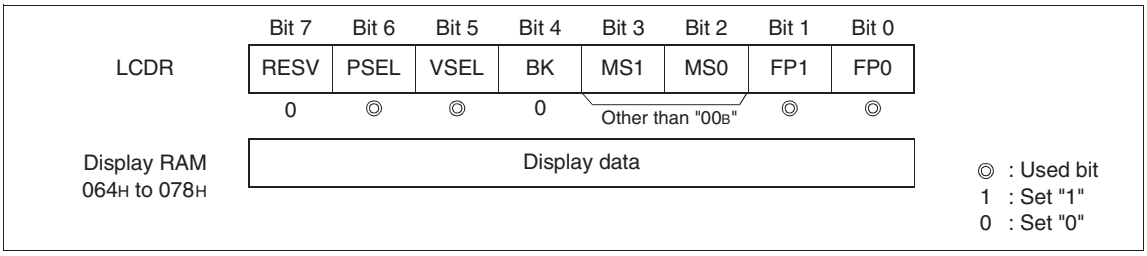
12.4 Operation of LCD Controller/Driver

The LCD controller/driver provides the necessary control and drive for an LCD.

■ Operation of LCD controller/driver

Figure 12.4-1 "LCD controller/driver settings" shows the settings required to operate the LCD.

Figure 12.4-1 LCD controller/driver settings



Once the above settings have been made, if the selected clock for frame cycle generation is running, LCD panel driving waveforms reflecting the contents of display RAM will be output at the segment and common output pins (COM0 to COM3 and SEG0 to SEG42).

Although the clock for frame period generation can be switched even while the LCD is displaying data, the display may flicker when the switching occurs. This can be avoided by temporarily blanking the display (LCDR: BK = "1"), etc. while switching.

The display driving output is a two-frame a.c. waveform for which the bias level and display duty cycle is selected by settings.

When LCD operation is stopped (LCDR: MS1, MS0 = "00_B"), and during reset, all COM and SEG output pins are pulled "L" state so that nothing is displayed on the LCD panel.

Note:

If the selected frame cycle generate clock were to stop while the LCD is operating, the circuit that converts the waveform from d.c. to a.c. would also stop, causing a d.c. voltage to be applied to the liquid crystal cells. The LCD must therefore be stopped before the clock is stopped. The conditions under which the main clock is stopped is a function of the clock mode and standby mode.

■ LCD driving waveforms

It is characteristic of LCD that applying d.c. drive to the panel can cause electrochemical degradation of the material used in the LCD cells. For this reason, the LCD controller/driver includes a circuit to convert the original driving waveform to a two-frame a.c. output waveform (zero d.c. bias) to drive the LCD. There are three types of output waveform:

- 1/2 bias, 1/2 duty ratio output waveform
- 1/3 bias, 1/3 duty ratio output waveform
- 1/3 bias, 1/4 duty ratio output waveform

12.4.1 Output Waveforms during LCD Controller/Driver Operation (1/2 Duty Ratio)

The display drive output is a multiplex drive-type two-frame a.c. waveform. In the 1/2 duty ratio mode, the only common outputs are COM0 and COM1. (COM2 and COM3 are not used.)

■ 1/2 bias, 1/2 duty output waveform

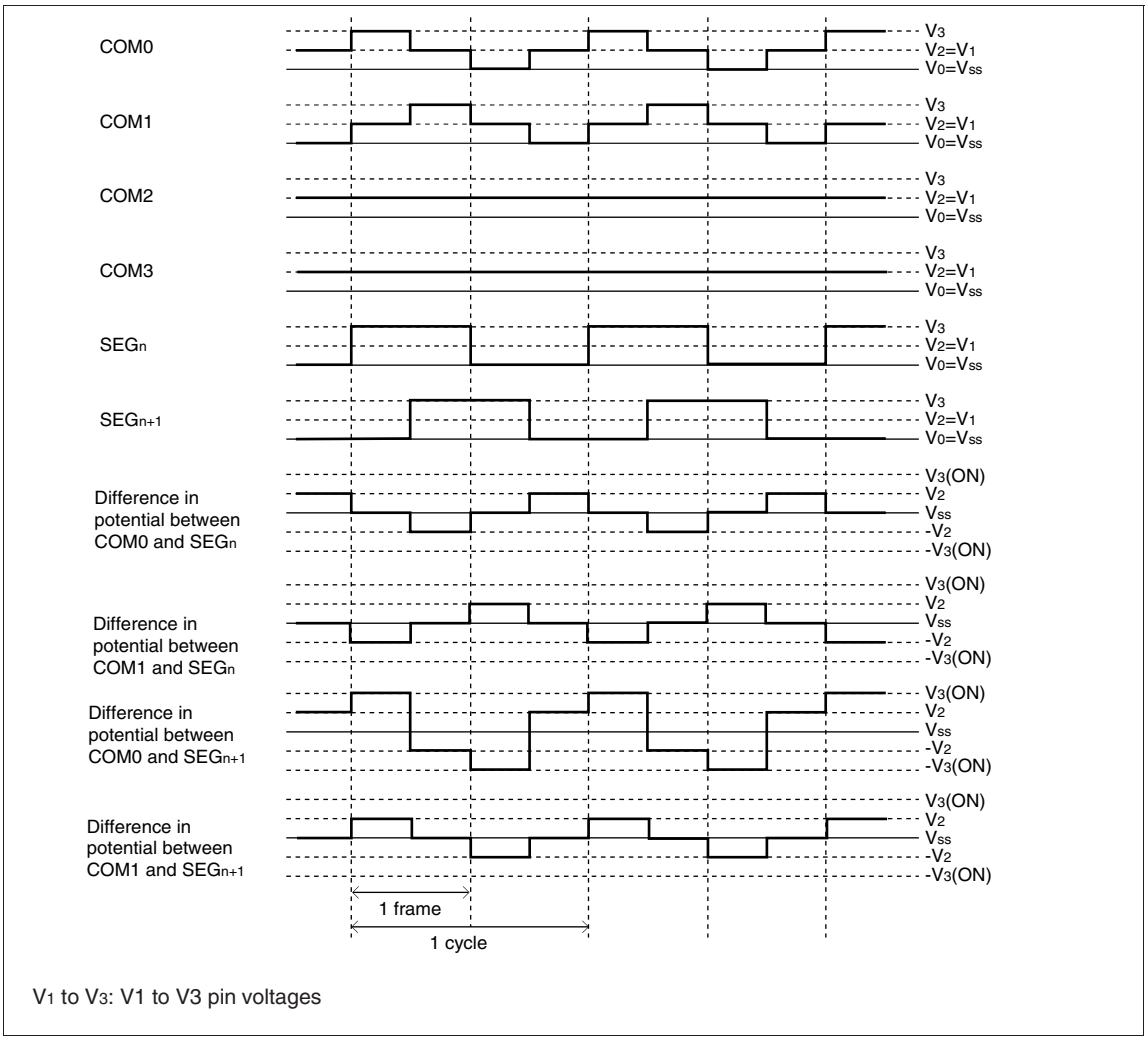
The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is turned on. Figure 12.4-2 "Output waveforms, 1/2 bias and 1/2 duty ratio example" shows the output waveforms for the display RAM contents listed in Table 12.4-1 "Display RAM contents example".

Table 12.4-1 Display RAM contents example

Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	–	–	0	0
SEGn+1	–	–	0	1

–: Not used

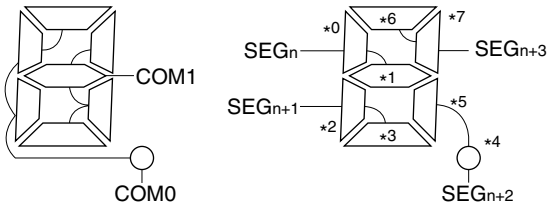
Figure 12.4-2 Output waveforms, 1/2 bias and 1/2 duty ratio example



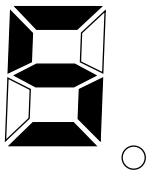
V1 to V3: V1 to V3 pin voltages

● LCD panel connections and display data example (1/2 duty ratio drive mode)

Figure 12.4-3 Segment/common connections, data states and corresponding display



Example) Using segments to represent "5".

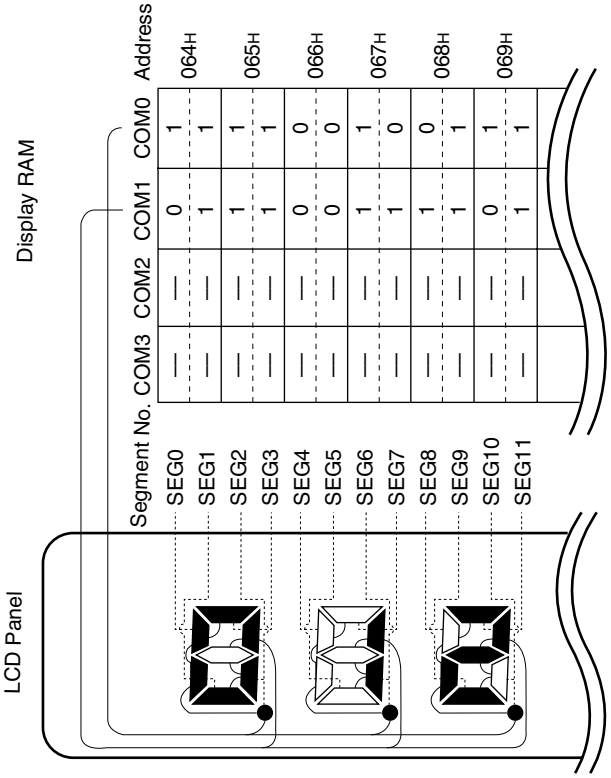


Address	COM3	COM2	COM1	COM0	
nH	bit3	bit2	bit1*1	bit0*0	SEGn
	bit7	bit6	bit5*3	bit4*2	SEGn+1
n+1H	bit3	bit2	bit1*5	bit0*4	SEGn+2
	bit7	bit6	bit5*7	bit4*6	SEGn+3

*0 to *7: Indicate corresponding display RAM bits. (Bits 2, 3, 6, and 7 are not used.)

Address	COM3	COM2	COM1	COM0	
064H	—	—	1	1	SEG0
	—	—	1	0	SEG1
065H	—	—	1	0	SEG2
	—	—	0	1	SEG3

0: OFF
1: ON



LCD Display	Bit States for Numerals "0" through "9"							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0	—	—	1	1	—	—	0	1
	—	—	1	1	—	—	1	1
1	—	—	0	0	—	—	0	0
	—	—	1	0	—	—	1	1
2	—	—	1	1	—	—	1	0
	—	—	1	1	—	—	0	1
3	—	—	1	1	—	—	1	0
	—	—	1	1	—	—	1	1
4	—	—	0	0	—	—	1	1
	—	—	1	0	—	—	1	1
5	—	—	1	0	—	—	1	1
	—	—	0	1	—	—	1	1
6	—	—	1	1	—	—	1	1
	—	—	0	1	—	—	1	1
7	—	—	0	0	—	—	0	1
	—	—	1	1	—	—	1	1
8	—	—	1	1	—	—	1	1
	—	—	1	1	—	—	1	1
9	—	—	1	0	—	—	1	1
	—	—	1	1	—	—	1	1

12.4.2 Output Waveforms during LCD Controller/Driver Operation (1/3 Duty Ratio)

In the 1/3 duty ratio mode, the COM0, COM1 and COM2 outputs are used by the display. COM3 is not used.

■ 1/3 bias, 1/3 duty output waveform

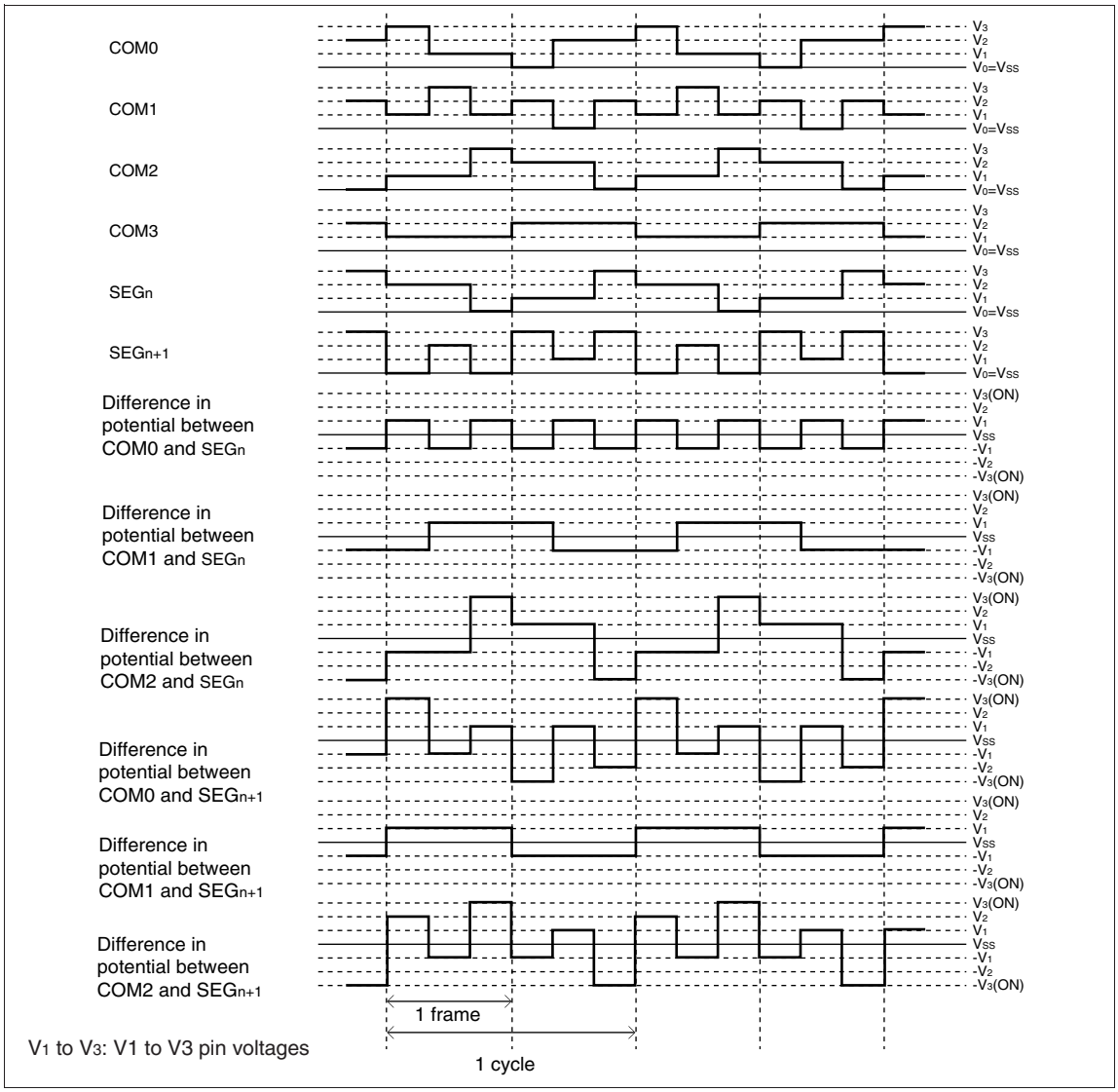
The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is turned on. Figure 12.4-4 "Output waveforms, 1/3 bias and 1/3 duty ratio example" shows the output waveforms for the display RAM contents listed in Table 12.4-2 "Display RAM contents example".

Table 12.4-2 Display RAM contents example

Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	–	1	0	0
SEGn+1	–	1	0	1

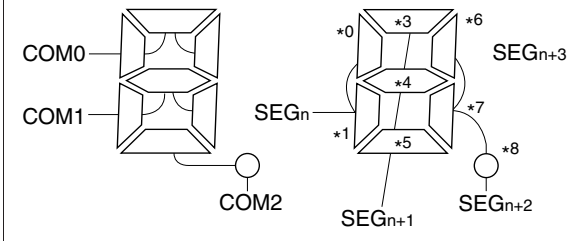
–: Not used

Figure 12.4-4 Output waveforms, 1/3 bias and 1/3 duty ratio example



● LCD panel connections and display data example (1/3 duty ratio drive mode)

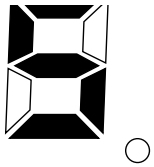
Figure 12.4-5 Segment/common connections, data states and corresponding display



Address	COM3	COM2	COM1	COM0	
nH	bit3	bit2*2	bit1*1	bit0*0	SEGn
	bit7	bit6*5	bit5*4	bit4*3	SEGn+1
n+1H	bit3	bit2*8	bit1*7	bit0*6	SEGn+2

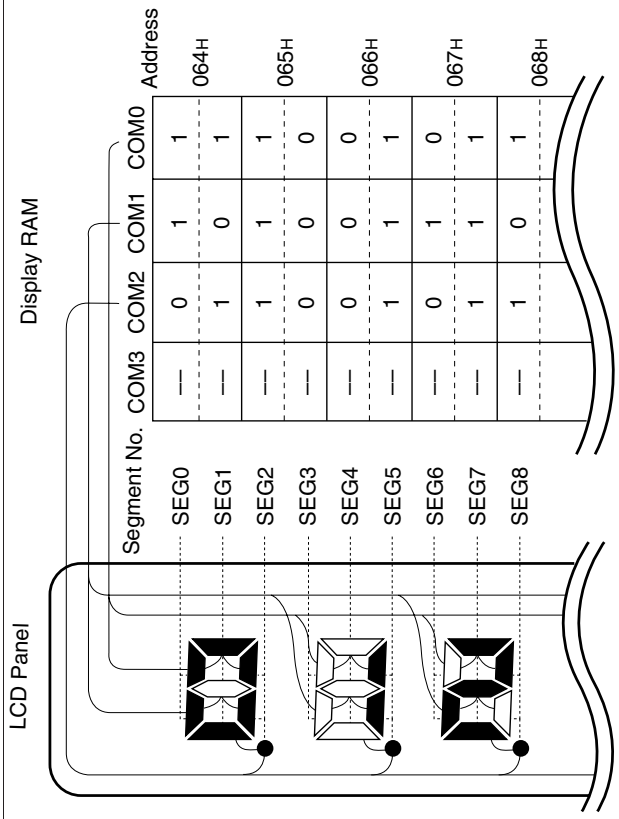
*0 to *8: Indicate corresponding display RAM bits. (Bits 3 and 7 and *2 are not used.)

Example) Using segments to represent "5".



Address	COM3	COM2	COM1	COM0	
064H	—	0	0	1	SEG0
	—	1	1	1	SEG1
065H	—	0	1	0	SEG2
	—	0	0	1	SEG3
066H	—	1	1	1	SEG4
	—	0	1	0	SEG5

0: OFF
1: ON



LCD Display	Bit States for Numerals "0" through "9"							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0.	—	1	0	1	—	0	1	1
	—	0	1	1	—	1	1	1
	—	1	1	1	—	1	0	1
1.	—	0	0	0	—	0	0	0
	—	0	0	0	—	1	1	1
	—	1	1	1	—	0	0	0
2.	—	1	1	1	—	0	1	0
	—	0	1	0	—	1	0	1
	—	1	0	1	—	1	1	1
3.	—	1	1	1	—	0	0	0
	—	0	0	0	—	1	1	1
	—	1	1	1	—	1	1	1
4.	—	0	1	0	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	0	1	0
5.	—	1	1	1	—	0	0	1
	—	0	0	1	—	1	1	0
	—	1	1	0	—	1	1	1
6.	—	1	1	1	—	0	1	1
	—	0	1	1	—	1	1	0
	—	1	1	0	—	1	1	1
7.	—	0	0	1	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	0	0	1
8.	—	1	1	1	—	0	1	1
	—	0	1	1	—	1	1	1
	—	1	1	1	—	1	1	1
9.	—	1	1	1	—	0	0	1
	—	0	0	1	—	1	1	1
	—	1	1	1	—	1	1	1

: Data in unit starting at bit 4
: Data in unit starting at bit 0

In 1/3 duty ratio operation, to be able to define two digits in three bytes, the data stored in two bytes, with the first byte starting at bit 0, and second byte starting at bit 4.

12.4.3 Output Waveforms during LCD Controller/Driver Operation (1/4 Duty Ratio)

In the 1/4 duty ratio mode, all four common outputs, COM0, COM1, COM2, and COM3 are used.

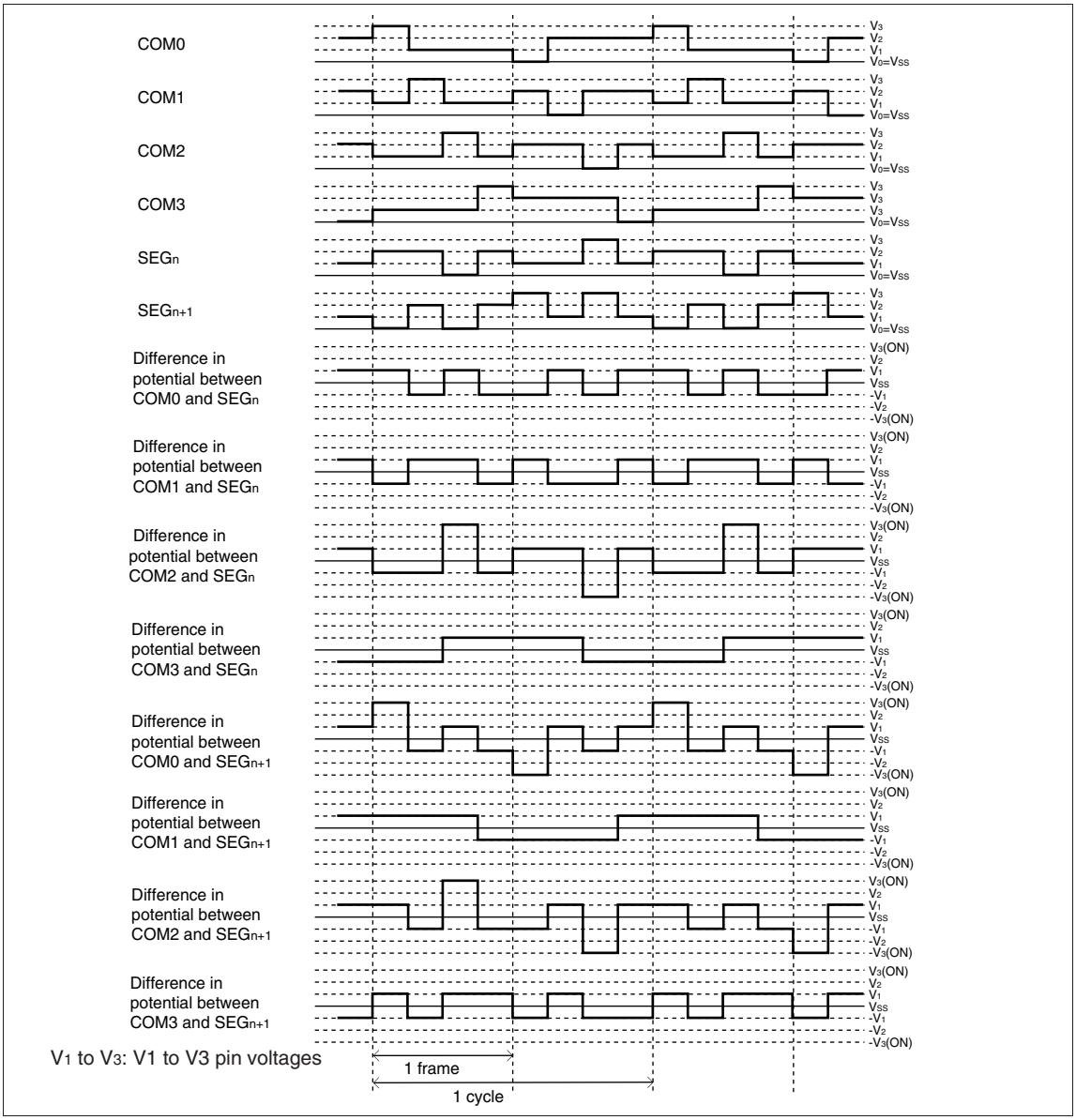
■ 1/3 bias, 1/4 duty output waveforms

The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is turned on. Figure 12.4-6 "Output waveforms, 1/3 bias and 1/4 duty ratio example" shows the output waveforms for the display RAM contents listed in Table 12.4-3 "Display RAM contents example".

Table 12.4-3 Display RAM contents example

Segment	Display RAM contents			
	COM3	COM2	COM1	COM0
SEGn	0	1	0	0
SEGn+1	0	1	0	1

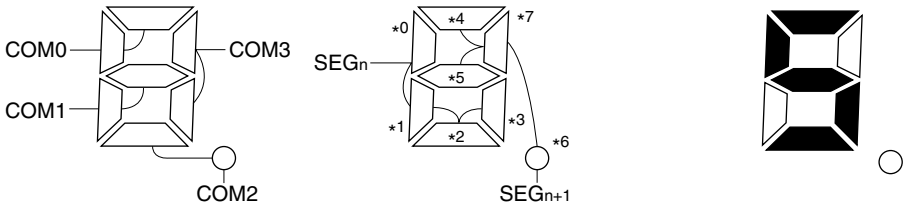
Figure 12.4-6 Output waveforms, 1/3 bias and 1/4 duty ratio example



● 8-segment LCD panel connections and display data (1/4 duty ratio drive mode)

Figure 12.4-7 Segment/common connections, data states and corresponding display

Example) Using segments to represent "5".

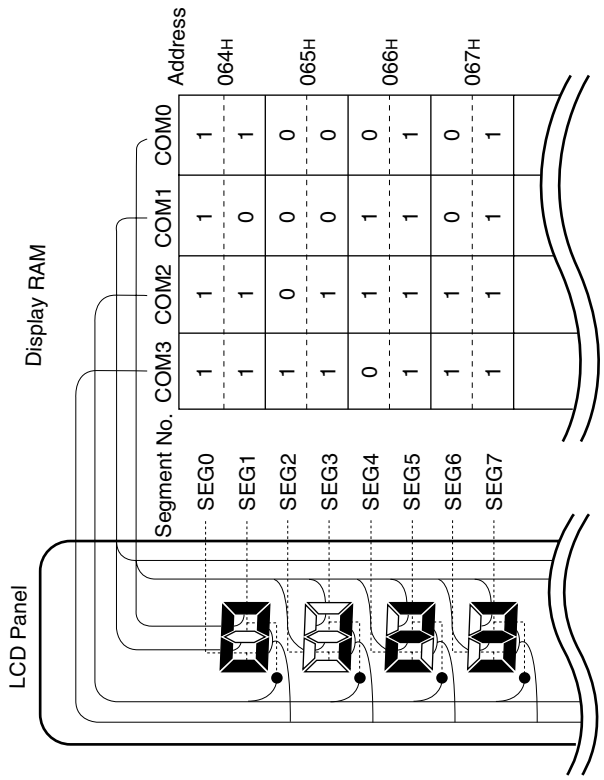


Address	COM3	COM2	COM1	COM0	
nH	bit3*3	bit2*2	bit1*1	bit0*0	SEGn
	bit7*7	bit6*6	bit5*5	bit4*4	SEGn+1

*0 to *7: Indicate corresponding display RAM bits.

Address	COM3	COM2	COM1	COM0	
064H	1	1	0	1	SEG0
	0	0	1	1	SEG1

0: OFF
1: ON



LCD Display	Bit States for Numerals "0" through "9"							
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0.	1	1	0	1	1	1	1	1
1.	1	1	0	0	1	0	0	0
2.	1	1	1	1	0	1	1	0
3.	1	1	1	1	1	1	0	0
4.	1	1	1	0	1	0	0	1
5.	0	1	1	1	1	1	0	1
6.	0	1	1	1	1	1	1	1
7.	1	1	0	1	1	0	0	1
8.	1	1	1	1	1	1	1	1
9.	1	1	1	1	1	1	0	1

12.5 Program Example for LCD Controller/Driver

This section gives a program example for LCD controller/driver.

■ Program example for LCD controller/driver

● Processing description

The process writes LCD data to display RAM. The data is that required to display the numbers "0" through "9" in an LCD panel connected as shown in Figure 12.4-7 "Segment/common connections, data states and corresponding display". The settings are as follows:

- Internal voltage divider is selected (LCDR: VSEL = "1")
- 1/3 bias and 1/4 duty ratio are used.
- The main clock oscillation frequency is 5 MHz
- The frame frequency is set at 76 Hz (LCDR: FP1, FP0 = "11_B")

● Coding example

```
LCRAM   EQU    0064H           ;Starting address of LCD display RAM
LCDR    EQU    0079H           ;Address of LCD control register (LCDR)
SEGR    EQU    007AH           ;Address of segment output select register (SEGR)
LCD     SEG    CSEG            ;8-segment LCD data
LCDDATA DB     11011111B       ;"0"
        DB     11001000B       ;"1"
        DB     11110110B       ;"2"
        DB     11111100B       ;"3"
        DB     11101001B       ;"4"
        DB     01111101B       ;"5"
        DB     01111111B       ;"6"
        DB     11011001B       ;"7"
        DB     11111111B       ;"8"
        DB     11111101B       ;"9"
        DB     00000000B       ;END
LCD     SEG    ENDS
;-----Main program-------------------------------------------------------------
        CSEG                   ; [CODE SEGMENT]
         :
        MOVW   EP,#LCRAM       ; Set LCD RAM address.
        MOVW   IX,#LCDDATA     ; Set LCD data table address.
        MOV    SEGR,01111111B  ; Set the segment output function.
LCDSET
        MOV    A,@IX+00H
        MOV    @EP,A
        INCW   EP
        INCW   IX
        BNZ    LCDSET          ; Continue until data end (00H) is detected.
        MOV    LCDR,#00101111B ; Set LCDR and turn LCD display on.
         :
        ENDS
;------------------------------------------------------------------------------
        END
```

APPENDIX

This appendix includes I/O maps, instruction lists, and other information.

APPENDIX A "I/O Map"

APPENDIX B "Overview of Instructions"

APPENDIX C "Mask Options"

APPENDIX D "Programming Specifications for One-Time PROM And EPROM Microcontroller"

APPENDIX E "MB89950/950A Series Pin States"

APPENDIX A I/O Map

Table A-1 "I/O map" lists the addresses of the registers of used by the internal peripheral functions of the MB89950/950A series.

■ I/O map

Table A-1 I/O map (1/2)

Address	Register name	Register description	Read/Write	Initial value
00_H	PDR0	Port 0 data register	R/W	11111111_B
01_H	(Vacancy)			
02_H	PDR1	Port 1 data register	R/W	11111111_B
03_H	(Vacancy)			
04_H	PDR2	Port 2 data register	R/W	$--111111_B$
05_H to 07_H	(Vacancy)			
08_H	STBC	Standby control register	R/W	$0001----_B$
09_H	WDTC	Watchdog timer control register	W	$----XXXX_B$
$0A_H$	TBTC	Timebase timer control register	R/W	$---00000_B$
$0B_H$	(Vacancy)			
$0C_H$	PDR3	Port 3 data register	R/W	$----1111_B$
$0D_H$	(Vacancy)			
$0E_H$	PDR4	Port 4 data register	R/W	$-XXXXXXX_B$
$0F_H$	DDR4	Port 4 direction register	W	-0000000_B
10_H to 11_H	(Vacancy)			
12_H	CNTR	PWM timer control register	R/W	$0-000000_B$
13_H	COMR	PWM timer compare register	W	$XXXXXXXX_B$
14_H	PCR1	PWC pulse width control register 1	R/W	$0-0--000_B$
15_H	PCR2	PWC pulse width control register 2	R/W	$000-0000_B$
16_H	RLBR	PWC reload buffer register	R/W	$XXXXXXXX_B$
17_H	NCCR	PWC noise filter control register	R/W	$------00_B$
18_H to $1B_H$	(Vacancy)			
$1C_H$	SMR	Serial mode register	R/W	00000000_B

Table A-1 I/O map (2/2)

Address	Register name	Register description	Read/Write	Initial value
$1D_H$	SDR	Serial data register	R/W	$XXXXXXXX_B$
$1E_H$	(Vacancy)			
$1F_H$				
20_H	SMC1	UART serial mode control register 1	R/W	$00000\text{-}00_B$
21_H	SRC	UART serial rate control register	R/W	$\text{--}011000_B$
22_H	SSD	UART serial status/data register	R/W	$00100\text{-}1X_B$
23_H	SIDR/SODR	UART serial data register	R/W	$XXXXXXXX_B$
24_H	SMC2	UART serial mode control register 2	R/W	$\text{--}1\text{-}0\text{-}00_B$
25_H to $2F_H$	(Vacancy)			
30_H	EIC	External interrupt control register	R/W	00000000_B
31_H to 63_H	(Vacancy)			
64_H to 78_H	VRAM	LCD data RAM	R/W	$XXXXXXXX_B$
79_H	LCDR	LCD control register	R/W	$\text{-}0010000_B$
$7A_H$	SEGR	Segment output select register	R/W	$\text{-}0000000_B$
$7B_H$	(Vacancy)			
$7C_H$	ILR1	Interrupt level setting register 1	W	11111111_B
$7D_H$	ILR2	Interrupt level setting register 2	W	11111111_B
$7E_H$	ILR3	Interrupt level setting register 3	W	11111111_B
$7F_H$	ITR	Interrupt test register	Access prohibited	$XXXXXX00_B$

● Read/write access symbols

R/W: Readable and writable

R: Read-only

W: Write-only

● Initial value symbols

0: The initial value of this bit is "0"

1: The initial value of this bit is "1"

X: The initial value of this bit is undefined

-: Unused

Note:

Do not use vacancies.

APPENDIX B Overview of Instructions

Appendix B describes the instructions used by the F^2MC-8L.

B.1 "Overview of F^2MC-8L Instructions"

B.2 "Addressing"

B.3 "Special Instructions"

B.4 "Bit Manipulation Instructions (SETB, CLRB)"

B.5 "F^2MC-8L Instructions"

B.6 "Instruction Map"

B.1 Overview of F²MC-8L Instructions

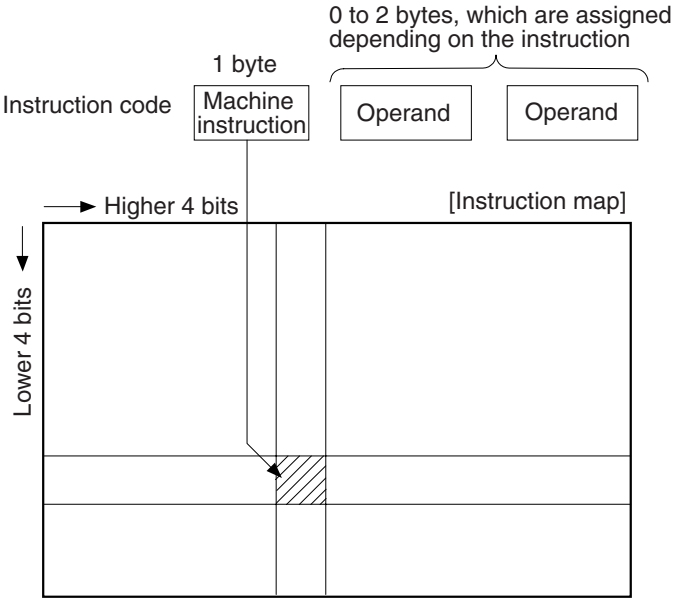
The F²MC-8L supports 140 types of instructions.

■ Overview of F²MC-8L instructions

The F²MC-8L has 140 1-byte machine instructions (256-byte instruction map). An instruction code consists of an instruction and zero or more operands that follow.

Figure B.1-1 "Relationship between the instruction codes and the instruction map" shows the relationship between the instruction codes and the instruction map.

Figure B.1-1 Relationship between the instruction codes and the instruction map



- The instructions are classified into four types: transfer, arithmetic, branch, and other.
- A variety of addressing methods is available. One of ten addressing modes can be selected depending on the selected instruction and specified operand(s).
- Bit manipulation instructions are provided. They can be used for read-modify-write operations.
- Some instructions are used for special operations.

■ Symbols used with Instructions

Table B.1-1 "Symbols in the instruction list" lists the symbols used in the instruction code descriptions in Appendix B.

Table B.1-1 Symbols in the instruction list

Symbol	Meaning
dir	Direct address (8 bits)
off	Offset (8 bits)
ext	Extended address (16 bits)
#vct	Vector table number (3 bits)
#d8	Immediate data (8 bits)
#d16	Immediate data (16 bits)
dir:16	Bit direct address (8 bits:3 bits)
rel	Branch relative address (8 bits)
@	Register indirect addressing (examples: @A, @IX, @EP)
A	Accumulator (8 or 16 bits, which are determined depending on the instruction being used)
AH	Higher 8 bits of the accumulator (8 bits)
AL	Lower 8 bits of the accumulator (8 bits)
T	Temporary accumulator (8 or 16 bits, which are determined depending on the instruction being used)
TH	Higher 8 bits of the temporary accumulator (8 bits)
TL	Lower 8 bits of the temporary accumulator (8 bits)
IX	Index register (16 bits)
EP	Extra pointer (16 bits)
PC	Program counter (16 bits)
SP	Stack pointer (16 bits)
PS	Program status (16 bits)
dr	Either accumulator or index register (16 bits)
CCR	Condition code register (8 bits)
RP	Register bank pointer (5 bits)
Ri	General-purpose register (8 bits, i = 0 to 7)
X	X is immediate data (8 or 16 bits, which are determined depending on the instruction being used).
(X)	The content of X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).
((X))	The address indicated by the X is to be accessed (8 or 16 bits, which are determined depending on the instruction being used).

B.2 Addressing

The F^2MC-8L has the following ten addressing modes:

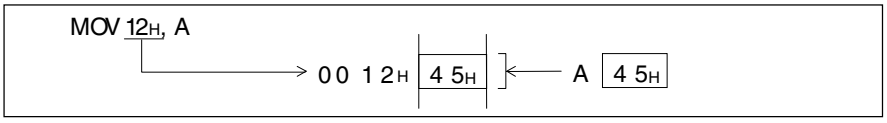
- **Direct addressing**
- **Extended addressing**
- **Bit direct addressing**
- **Index addressing**
- **Pointer addressing**
- **General-purpose register addressing**
- **Immediate addressing**
- **Vector addressing**
- **Relative addressing**
- **Inherent addressing**

■ Explanation of addressing

● Direct addressing

Direct addressing is indicated by dir in the instruction list. This addressing is used to access the area between 0000_H and $00FF_H$. In this addressing mode, the higher byte of the address is 00_H and the lower byte is specified by the operand. Figure B.2-1 "Example of direct addressing" shows an example.

Figure B.2-1 Example of direct addressing

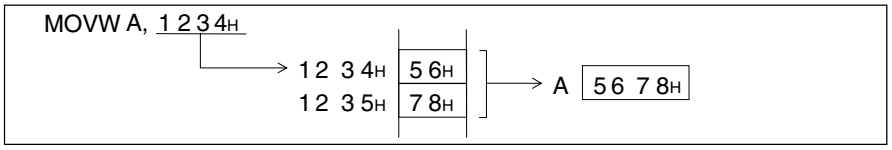


● Extended addressing

Extended addressing is indicated by ext in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the first operand specifies the higher byte of the address, and the second operand specifies the lower byte.

Figure B.2-2 "Example of Extended Addressing" shows an example.

Figure B.2-2 Example of extended addressing

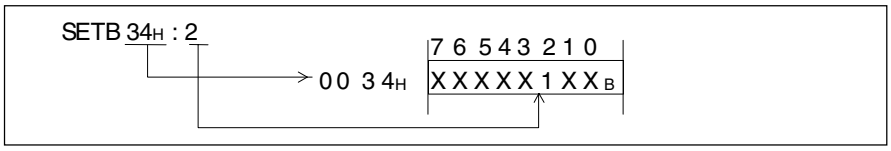


● Bit direct addressing

Bit direct addressing is indicated by dir:b in the instruction list. This addressing is used to access a particular bit in the area between 0000_H and $00FF_H$. In this addressing mode, the higher byte of the address is 00_H and the lower byte is specified by the operand. The bit position at the address is specified by the lower three bits of the operation code.

Figure B.2-3 "Example of bit direct addressing" shows an example.

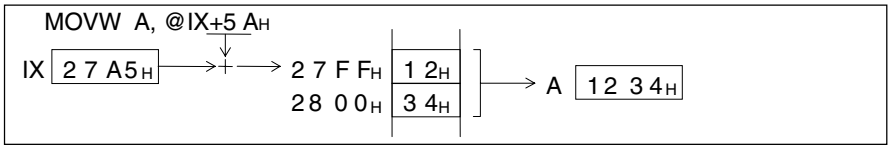
Figure B.2-3 Example of bit direct addressing



● Index addressing

Index addressing is indicated by @IX+off in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is the value resulting from sign-extending the contents of the first operand and adding them to IX (index register). Figure B.2-4 "Example of index addressing" shows an example.

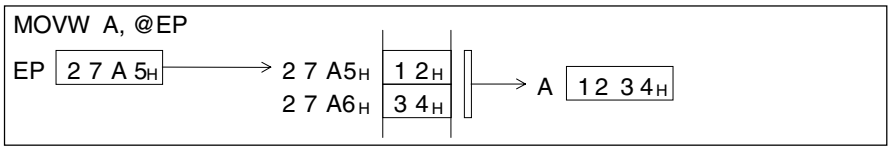
Figure B.2-4 Example of index addressing



● Pointer addressing

Pointer addressing is indicated by @EP in the instruction list. This addressing is used to access the entire 64-KB area. In this addressing mode, the address is contained in EP (extra pointer). Figure B.2-5 "Example of pointer addressing" shows an example.

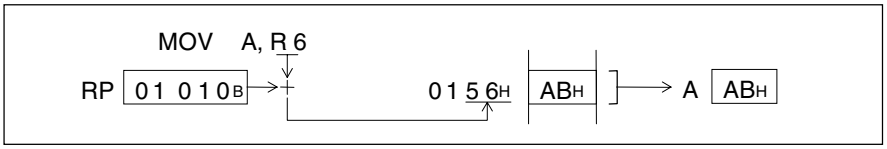
Figure B.2-5 Example of pointer addressing



● General-purpose register addressing

General-purpose register addressing is indicated by Ri in the instruction list. This addressing is used to access a register bank in the general-purpose register area. In this addressing mode, the higher byte of the address is always 01 and the lower byte is specified based on the contents of RP (register bank pointer) and the lower three bits of the operation code. Figure B.2-6 "Example of general-purpose register addressing" shows an example.

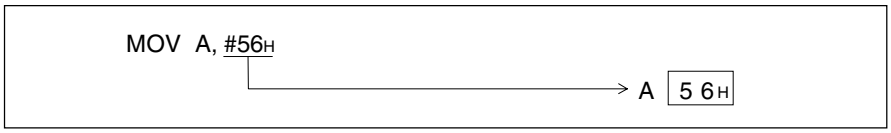
Figure B.2-6 Example of general-purpose register addressing



● Immediate addressing

Immediate addressing is indicated by #d8 in the instruction list. This addressing is used when immediate data is required. In this addressing mode, the operand is used as immediate data. Whether the data is specified in bytes or words is determined by the operation code. Figure B.2-7 "Example of immediate addressing" shows an example.

Figure B.2-7 Example of immediate addressing



● Vector addressing

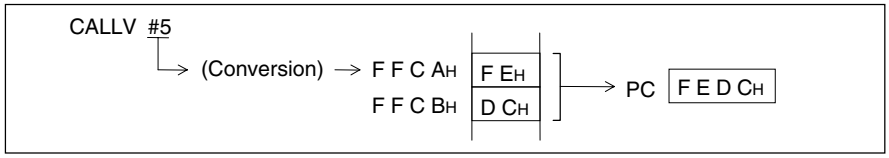
Vector addressing is indicated by vct in the instruction list. This addressing is used to branch to a subroutine address stored in the vector table. In this addressing mode, vct information is contained in the operation codes, and the corresponding table addresses are created as shown in Table B.2-1 "Vector table addresses corresponding to vct".

Table B.2-1 Vector table addresses corresponding to vct

#vct	Vector table address (higher address:lower address of branch destination)
0	$FFC0_H : FFC1_H$
1	$FFC2_H : FFC3_H$
2	$FFC4_H : FFC5_H$
3	$FFC6_H : FFC7_H$
4	$FFC8_H : FFC9_H$
5	$FFCA_H : FFCB_H$
6	$FFCC_H : FFCD_H$
7	$FFCE_H : FFCF_H$

Figure B.2-8 "Example of vector addressing" shows an example.

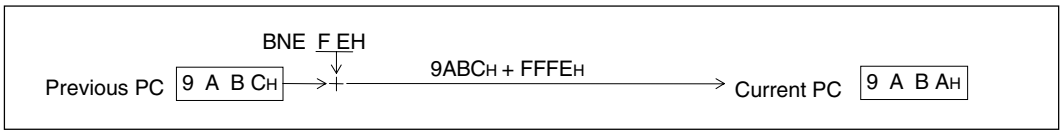
Figure B.2-8 Example of vector addressing



● Relative addressing

Relative addressing is indicated by rel in the instruction list. This addressing is used to branch to within the area between the address 128 bytes higher and that 128 bytes lower relative to the address contained in the PC (program counter). In this addressing mode, the result of a signed addition of the contents of the operand to the PC is stored in the PC. Figure B.2-9 "Example of relative addressing" shows an example.

Figure B.2-9 Example of relative addressing

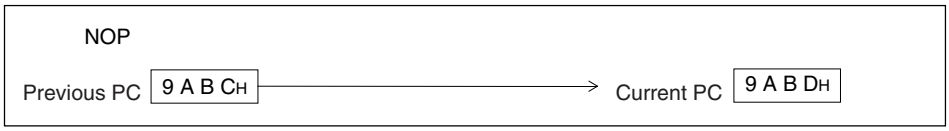


In this example, a branch to the address of the BNE operation code occurs, thus resulting in an infinite loop.

● Inherent addressing

Inherent addressing is indicated as the addressing without operands in the instruction list. This addressing is used to perform the operation determined by the operation code. In this addressing mode, different operations are performed via different instructions. Figure B.2-10 "Example of inherent addressing" shows an example.

Figure B.2-10 Example of inherent addressing



B.3 Special Instructions

This section describes the special instructions used for other than addressing.

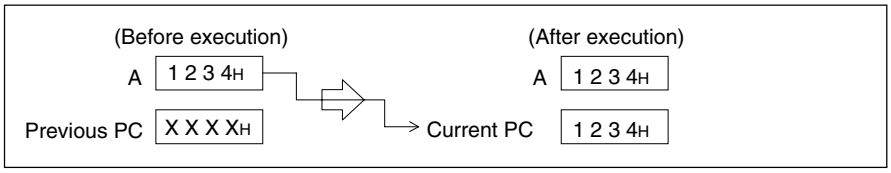
■ Special instructions

● JMP @A

This instruction sets the contents of A (accumulator) to PC (program counter) as the address, and causes a branch to that address. One of the N branch destination addresses is selected from a table, and then transferred to A. The instruction can be executed to perform N-branch processing.

Figure B.3-1 "JMP @A" shows a summary of the instruction.

Figure B.3-1 JMP @A

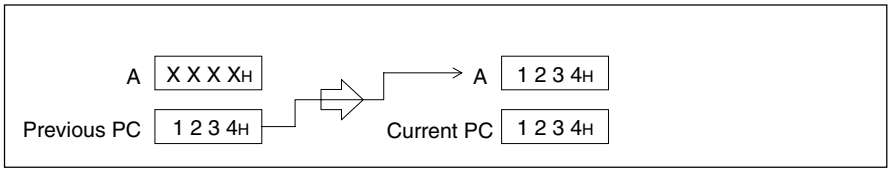


● MOVW A, PC

This instruction performs the operation which is the reverse of that performed by JMP @A. That is, the instruction stores the contents of PC in A. When the instruction is executed in the main routine, so that a specific subroutine is called, whether A contains a predetermined value can be checked by the subroutine. This can be used to determine that the branch source is not any unexpected section of the program and to check for program runaway.

Figure B.3-2 "MOVW A, PC" shows a summary of the instruction.

Figure B.3-2 MOVW A, PC



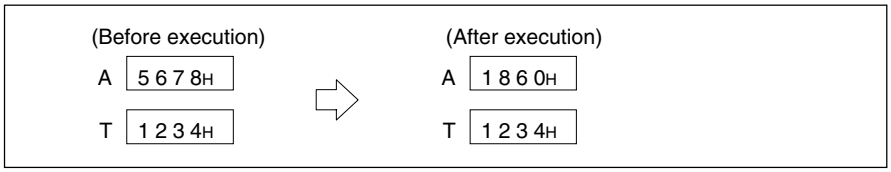
After the MOVW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of MOVW A, PC. Accordingly, Figure B.3-2 "MOVW A, PC" shows that A contains 1234_H, which is the address of the operation code of the instruction that follows MOVW A, PC.

● MULU A

This instruction performs an unsigned multiplication of AL (lower eight bits of the accumulator) and TL (lower eight bits of the temporary accumulator), and stores the 16-bit result in A. The contents of T (temporary accumulator) do not change. The contents of AH (higher eight bits of the accumulator) and TH (higher eight bits of the temporary accumulator) before execution of the instruction are not used for the operation. The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a multiplication.

Figure B.3-3 "MULU" shows a summary of the instruction.

Figure B.3-3 MULU



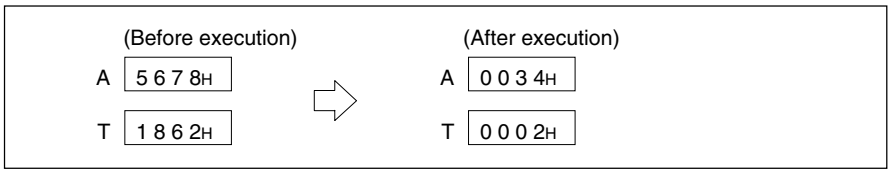
● DIVU A

This instruction divides the 16-bit value in T by the unsigned 8-bit value in AL, and stores the 8-bit result and the 8-bit remainder in AL and TL, respectively. A value of 0 is set to both AH and TH. The contents of AH before execution of the instruction are not used for the operation. An unpredictable result is produced from data that results in more than eight bits. In addition, there is no indication of the result having more than eight bits. Therefore, if it is likely that data will cause a result of more than eight bits, the data must be checked to ensure that the result will not have more than eight bits before it is used.

The instruction does not change the flags, and therefore care must be taken when a branch may occur depending on the result of a division.

Figure B.3-4 "DIVU A" shows a summary of the instruction.

Figure B.3-4 DIVU A

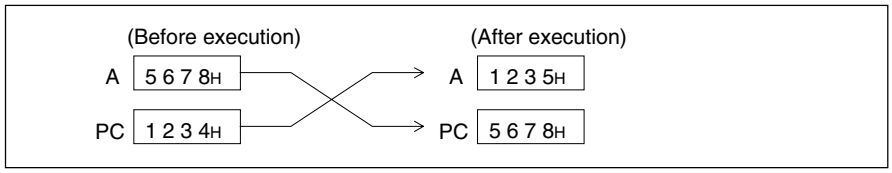


● XCHW A, PC

This instruction swaps the contents of A and PC, resulting in a branch to the address contained in A before execution of the instruction. After the instruction is executed, A contains the address that follows the address of the operation code of MOVW A, PC. This instruction is effective especially when it is used in the main routine to specify a table for use in a subroutine.

Figure B.3-5 "XCHW A, PC" shows a summary of the instruction.

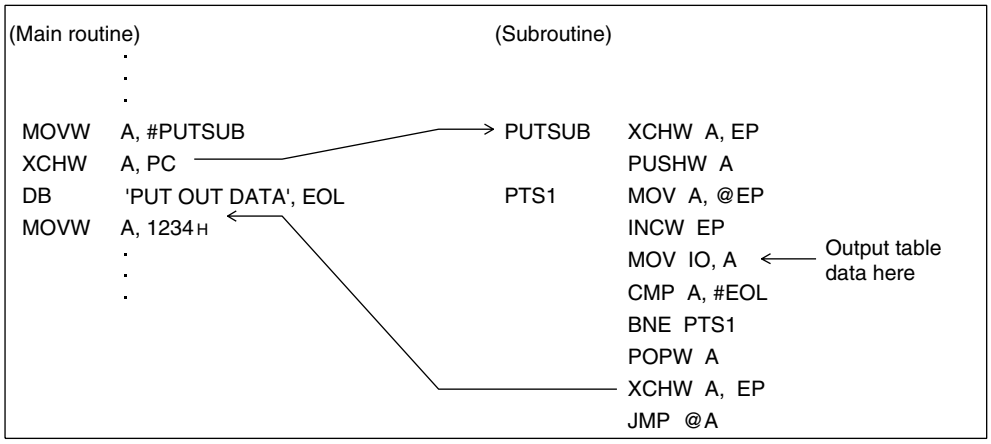
Figure B.3-5 XCHW A, PC



After the XCHW A, PC instruction is executed, A contains the address of the operation code of the next instruction, rather than the address of the operation code of XCHW A, PC. Accordingly, Figure B.3-5 "XCHW A, PC" shows that A contains 1235_H, which is the address of the operation code of the instruction that follows XCHW A, PC. This is why 1235_H is stored instead of 1234_H.

Figure B.3-6 "Example of using XCHW A, PC" shows an assembly language example.

Figure B.3-6 Example of using XCHW A, PC

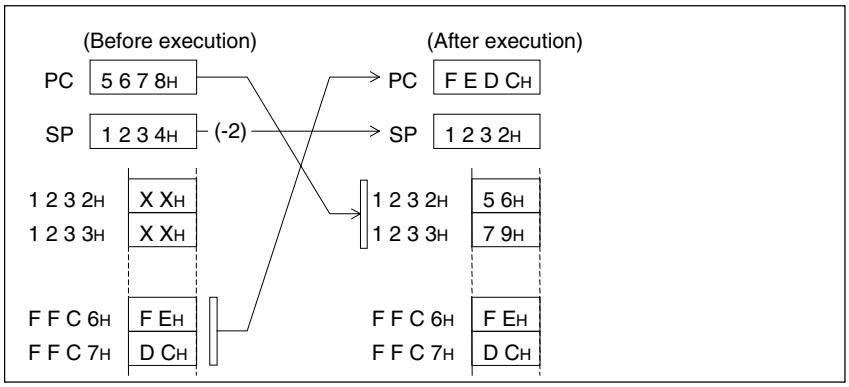


● CALLV #vct

This instruction is used to branch to a subroutine address stored in the vector table. The instruction saves the return address (contents of PC) in the location at the address contained in SP (stack pointer), and uses vector addressing to cause a branch to the address stored in the vector table. Because CALLV #vct is a 1-byte instruction, the use of this instruction for frequently used subroutines can reduce the entire program size.

Figure B.3-7 "Example of executing CALLV #3" shows a summary of the instruction.

Figure B.3-7 Example of executing CALLV #3



After the CALLV #vct instruction is executed, the contents of PC saved on the stack area are the address of the operation code of the next instruction, rather than the address of the operation code of CALLV #vct. Accordingly, Figure B.3-7 "Example of executing CALLV #3" shows that the value saved in the stack (1232_H and 1233_H) is 5679_H, which is the address of the operation code of the instruction that follows CALLV #vct (return address).

B.4 Bit Manipulation Instructions (SETB, CLRB)

Some bits of peripheral function registers include bits that are read by a bit manipulation instruction differently than usual.

■ Read-modify-write operation

By using these bit manipulation instructions, only the specified bit in a register or RAM location can be set to 1 (SETB) or cleared to 0 (CLRB). However, as the CPU operates on data in 8-bit units, the actual operation (read-modify-write operation) involves a sequence of steps: 8-bit data is read, the specified bit is changed, and the data is written back to the location at the original address.

Table B.4-1 "Bus operation for bit manipulation instructions" shows bus operation for bit manipulation instructions.

Table B.4-1 Bus operation for bit manipulation instructions

CODE	MNEMONIC	TO	Cycle	Address bus	Data bus	$\overline{RD}$	$\overline{WR}$	RMW
A0 to A7	CLRB dir:b	4	1	N+1	dir	0	1	0
			2	dir address	Data	0	1	1
A8 to AF	SETB dir:b		3	dir address	Data	1	0	0
			4	N+2	Next instruction	0	1	0

■ Read operation upon the execution of bit manipulation instructions

For some I/O ports and for the interrupt request flag bits, the value to be read differs between a normal read operation and a read-modify-write operation.

- I/O ports (during a bit manipulation)

From some I/O ports, an I/O pin value is read during a normal read operation, while an output latch value is read during a bit manipulation. This prevents the other output latch bits from being changed accidentally, regardless of the I/O directions and states of the pins.

- Interrupt request flag bits (during a bit manipulation)

An interrupt request flag bit functions as a flag bit indicating whether an interrupt request exists during a normal read operation. However, 1 is always read from this bit during a bit manipulation. This prevents the flag from being cleared accidentally by a value of 0 which would otherwise be written to the interrupt request flag bit when another bit is manipulated.

B.5 F²MC-8L Instructions

Table B.5-1 "Transfer instructions" to Table B.5-4 "Other instructions" list the instructions used with the F²MC-8L.

■ Transfer instructions

Table B.5-1 Transfer instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	MOV dir, A	3	2	(dir)<--(A)	-	-	-	-	-	-	-	45
2	MOV @IX+off, A	4	2	((IX)+off)<--(A)	-	-	-	-	-	-	-	46
3	MOV ext, A	4	3	(ext)<--(A)	-	-	-	-	-	-	-	61
4	MOV @EP, A	3	1	((EP))<--(A)	-	-	-	-	-	-	-	47
5	MOV Ri, A	3	1	(Ri)<--(A)	-	-	-	-	-	-	-	48 to 4F
6	MOV A, #d8	2	2	(A)<--d8	AL	-	-	+	+	-	-	04
7	MOV A, dir	3	2	(A)<--(dir)	AL	-	-	+	+	-	-	05
8	MOV A, @IX+off	4	2	(A)<--((IX)+off)	AL	-	-	+	+	-	-	06
9	MOV A, ext	4	3	(A)<--(ext)	AL	-	-	+	+	-	-	60
10	MOV A, @A	3	1	(A)<--((A))	AL	-	-	+	+	-	-	92
11	MOV A, @EP	3	1	(A)<--((EP))	AL	-	-	+	+	-	-	07
12	MOV A, Ri	3	1	(A)<--(Ri)	AL	-	-	+	+	-	-	08 to 0F
13	MOV dir, #d8	4	3	(dir)<--d8	-	-	-	-	-	-	-	85
14	MOV @IX+off, #d8	5	3	((IX)+off)<--d8	-	-	-	-	-	-	-	86
15	MOV @EP, #d8	4	2	((EP))<--d8	-	-	-	-	-	-	-	87
16	MOV Ri, #d8	4	2	(Ri)<--d8	-	-	-	-	-	-	-	88 to 8F
17	MOVW dir, A	4	2	(dir)<--(AH), (dir+1)<--(AL)	-	-	-	-	-	-	-	D5
18	MOVW @IX+off, A	5	2	((IX)+off)<--(AH), ((IX)+off+1)<--(AL)	-	-	-	-	-	-	-	D6
19	MOVW ext, A	5	3	(ext)<--(AH), (ext+1)<--(AL)	-	-	-	-	-	-	-	D4
20	MOVW @EP, A	4	1	((EP))<--(AH), ((EP)+1)<--(AL)	-	-	-	-	-	-	-	D7
21	MOVW EP, A	2	1	(EP)<--(A)	-	-	-	-	-	-	-	E3
22	MOVW A, #d16	3	3	(A)<--d16	AL	AH	dH	+	+	-	-	E4
23	MOVW A, dir	4	2	(AH)<--(dir), (AL)<--(dir+1)	AL	AH	dH	+	+	-	-	C5

Table B.5-1 Transfer instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
24	MOVW A, @IX+off	5	2	(AH)<--((IX)+off), (AL)<--((IX)+off+1)	AL	AH	dH	+	+	-	-	C6
25	MOVW A, ext	5	3	(AH)<--(ext), (AL)<--(ext+1)	AL	AH	dH	+	+	-	-	C4
26	MOVW A, @A	4	1	(AH)<--((A)), (AL)<--((A)+1)	AL	AH	dH	+	+	-	-	93
27	MOVW A, @EP	4	1	(AH)<--((EP)), (AL)<--((EP)+1)	AL	AH	dH	+	+	-	-	C7
28	MOVW A, EP	2	1	(A)<--(EP)	-	-	dH	-	-	-	-	F3
29	MOVW EP, #d16	3	3	(EP)<--d16	-	-	-	-	-	-	-	E7
30	MOVW IX, A	2	1	(IX)<--(A)	-	-	-	-	-	-	-	E2
31	MOVW A, IX	2	1	(A)<--(IX)	-	-	dH	-	-	-	-	F2
32	MOVW SP, A	2	1	(SP)<--(A)	-	-	-	-	-	-	-	E1
33	MOVW A, SP	2	1	(A)<--(SP)	-	-	dH	-	-	-	-	F1
34	MOV @A, T	3	1	((A))<--(T)	-	-	-	-	-	-	-	82
35	MOVW @A, T	4	1	((A))<--(TH), ((A)+1)<--(TL)	-	-	-	-	-	-	-	83
36	MOVW IX, #d16	3	3	(IX)<--d16	-	-	-	-	-	-	-	E6
37	MOVW A, PS	2	1	(A)<--(PS)	-	-	dH	-	-	-	-	70
38	MOVW PS, A	2	1	(PS)<--(A)	-	-	-	+	+	+	+	71
39	MOVW SP, #d16	3	3	(SP)<--d16	-	-	-	-	-	-	-	E5
40	SWAP	2	1	(AH)<-- -->(AL)	-	-	AL	-	-	-	-	10
41	SETB dir:b	4	2	(dir):b <--1	-	-	-	-	-	-	-	A8 to AF
42	CLRB dir:b	4	2	(dir):b <--0	-	-	-	-	-	-	-	A0 to A7
43	XCH A, T	2	1	(AL)<-- -->(TL)	AL	-	-	-	-	-	-	42
44	XCHW A, T	3	1	(A)<-- -->(T)	AL	AH	dH	-	-	-	-	43
45	XCHW A, EP	3	1	(A)<-- -->(EP)	-	-	dH	-	-	-	-	F7
46	XCHW A, IX	3	1	(A)<-- -->(IX)	-	-	dH	-	-	-	-	F6
47	XCHW A, SP	3	1	(A)<-- -->(SP)	-	-	dH	-	-	-	-	F5
48	MOVW A, PC	2	1	(A)<--(PC)	-	-	dH	-	-	-	-	F0

Note:

In automatic transfer to T during byte transfer to A, AL is transferred to TL.

If an instruction has two or more operands, they are assumed to be saved in the order indicated by MNEMONIC.

■ Arithmetic instructions

Table B.5-2 Arithmetic operation instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	ADDC A, Ri	3	1	(A)<--(A)+(Ri)+C	-	-	-	+	+	+	+	28 to 2F
2	ADDC A, #d8	2	2	(A)<--(A)+d8+C	-	-	-	+	+	+	+	24
3	ADDC A, dir	3	2	(A)<--(A)+(dir)+C	-	-	-	+	+	+	+	25
4	ADDC A, @IX+off	4	2	(A)<--(A)+((IX)+off)+C	-	-	-	+	+	+	+	26
5	ADDC A, @EP	3	1	(A)<--(A)+((EP))+C	-	-	-	+	+	+	+	27
6	ADDCW A	3	1	(A)<--(A)+(T)+C	-	-	dH	+	+	+	+	23
7	ADDC A	2	1	(AL)<--(AL)+(TL)+C	-	-	-	+	+	+	+	22
8	SUBC A, Ri	3	1	(A)<--(A)-(Ri)-C	-	-	-	+	+	+	+	38 to 3F
9	SUBC A, #d8	2	2	(A)<--(A)-d8-C	-	-	-	+	+	+	+	34
10	SUBC A, dir	3	2	(A)<--(A)-(dir)-C	-	-	-	+	+	+	+	35
11	SUBC A, @IX+off	4	2	(A)<--(A)-((IX)+off)-C	-	-	-	+	+	+	+	36
12	SUBC A, @EP	3	1	(A)<--(A)-((EP))-C	-	-	-	+	+	+	+	37
13	SUBCW A	3	1	(A)<--(T)-(A)-C	-	-	dH	+	+	+	+	33
14	SUBC A	2	1	(AL)<--(TL)-(AL)-C	-	-	-	+	+	+	+	32
15	INC Ri	4	1	(Ri)<--(Ri)+1	-	-	-	+	+	+	-	C8 to CF
16	INCW EP	3	1	(EP)<--(EP)+1	-	-	-	-	-	-	-	C3
17	INCW IX	3	1	(IX)<--(IX)+1	-	-	-	-	-	-	-	C2
18	INCW A	3	1	(A)<--(A)+1	-	-	dH	+	+	-	-	C0
19	DEC Ri	4	1	(Ri)<--(Ri)-1	-	-	-	+	+	+	-	D8 to DF
20	DECW EP	3	1	(EP)<--(EP)-1	-	-	-	-	-	-	-	D3
21	DECW IX	3	1	(IX)<--(IX)-1	-	-	-	-	-	-	-	D2
22	DECW A	3	1	(A)<--(A)-1	-	-	dH	+	+	-	-	D0
23	MULU A	19	1	(A)<--(AL)x(TL)	-	-	dH	-	-	-	-	01
24	DIVU A	21	1	(A)<--(T)/(AL), MOD -->(T)	dL	00	00	-	-	-	-	11
25	ANDW A	3	1	(A)<--(A) ∧ (T)	-	-	dH	+	+	R	-	63
26	ORW A	3	1	(A)<--(A) ∨ (T)	-	-	dH	+	+	R	-	73
27	XORW A	3	1	(A)<--(A) ∀ (T)	-	-	dH	+	+	R	-	53
28	CMP A	2	1	(TL)-(AL)	-	-	-	+	+	+	+	12

Table B.5-2 Arithmetic operation instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
29	CMPW A	3	1	(T)-(A)	-	-	-	+	+	+	+	13
30	RORC A	2	1	→ C --> A ┐	-	-	-	+	+	-	+	03
31	ROLC A	2	1	┌ C <-- A ←	-	-	-	+	+	-	+	02
32	CMP A, #d8	2	2	(A)-d8	-	-	-	+	+	+	+	14
33	CMP A, dir	3	2	(A)-(dir)	-	-	-	+	+	+	+	15
34	CMP A, @EP	3	1	(A)-((EP))	-	-	-	+	+	+	+	17
35	CMP A, @IX+off	4	2	(A)-((IX)+off)	-	-	-	+	+	+	+	16
36	CMP A, Ri	3	1	(A)-(Ri)	-	-	-	+	+	+	+	18 to 1F
37	DAA	2	1	decimal adjust for addition	-	-	-	+	+	+	+	84
38	DAS	2	1	decimal adjust for subtraction	-	-	-	+	+	+	+	94
39	XOR A	2	1	(A)<--(AL) ∀ (TL)	-	-	-	+	+	R	-	52
40	XOR A, #d8	2	2	(A)<--(AL) ∀ d8	-	-	-	+	+	R	-	54
41	XOR A, dir	3	2	(A)<--(AL) ∀ (dir)	-	-	-	+	+	R	-	55
42	XOR A, @EP	3	1	(A)<--(AL) ∀ ((EP))	-	-	-	+	+	R	-	57
43	XOR A, @IX+off	4	2	(A)<--(AL) ∀ ((IX)+off)	-	-	-	+	+	R	-	56
44	XOR A, Ri	3	1	(A)<--(AL) ∀ (Ri)	-	-	-	+	+	R	-	58 to 5F
45	AND A	2	1	(A)<--(AL) ∧ (TL)	-	-	-	+	+	R	-	62
46	AND A, #d8	2	2	(A)<--(AL) ∧ d8	-	-	-	+	+	R	-	64
47	AND A, dir	3	2	(A)<--(AL) ∧ (dir)	-	-	-	+	+	R	-	65
48	AND A, @EP	3	1	(A)<--(AL) ∧ ((EP))	-	-	-	+	+	R	-	67
49	AND A, @IX+off	4	2	(A)<--(AL) ∧ ((IX)+off)	-	-	-	+	+	R	-	66
50	AND A, Ri	3	1	(A)<--(AL) ∧ (Ri)	-	-	-	+	+	R	-	68 to 6F
51	OR A	2	1	(A)<--(AL) ∨ (TL)	-	-	-	+	+	R	-	72
52	OR A, #d8	2	2	(A)<--(AL) ∨ d8	-	-	-	+	+	R	-	74
53	OR A, dir	3	2	(A)<--(AL) ∨ (dir)	-	-	-	+	+	R	-	75

Table B.5-2 Arithmetic operation instructions (Continued)

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
54	OR A, @EP	3	1	(A)<--(AL) ∨ ((EP))	-	-	-	+	+	R	-	77
55	OR A, @IX+off	4	2	(A)<--(AL) ∨ ((IX)+off)	-	-	-	+	+	R	-	76
56	OR A, Ri	3	1	(A)<--(AL) ∨ (Ri)	-	-	-	+	+	R	-	78 to 7F
57	CMP dir, #d8	5	3	(dir)-d8	-	-	-	+	+	+	+	95
58	CMP @EP, #d8	4	2	((EP))-d8	-	-	-	+	+	+	+	97
59	CMP @IX+off, #d8	5	3	((IX)+off)-d8	-	-	-	+	+	+	+	96
60	CMP Ri, #d8	4	2	(Ri)-d8	-	-	-	+	+	+	+	98 to 9F
61	INCW SP	3	1	(SP)<--(SP)+1	-	-	-	-	-	-	-	C1
62	DECW SP	3	1	(SP)<--(SP)-1	-	-	-	-	-	-	-	D1

■ Branch instructions

Table B.5-3 Branch instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	BZ/BEQ rel	3	2	if Z=1 then PC<--PC+rel	-	-	-	-	-	-	-	FD
2	BNZ/BNE rel	3	2	if Z=0 then PC<--PC+rel	-	-	-	-	-	-	-	FC
3	BC/BLO rel	3	2	if C=1 then PC<--PC+rel	-	-	-	-	-	-	-	F9
4	BNC/BHS rel	3	2	if C=0 then PC<--PC+rel	-	-	-	-	-	-	-	F8
5	BN rel	3	2	if N=1 then PC<--PC+rel	-	-	-	-	-	-	-	FB
6	BP rel	3	2	if N=0 then PC<--PC+rel	-	-	-	-	-	-	-	FA
7	BLT rel	3	2	if V ∀ N=1 then PC<--PC+rel	-	-	-	-	-	-	-	FF
8	BGE rel	3	2	if V ∀ N=0 then PC<--PC+rel	-	-	-	-	-	-	-	FE
9	BBC dir:b, rel	5	3	if (dir:b)=0 then PC<--PC+rel	-	-	-	-	+	-	-	B0 to B7
10	BBS dir:b, rel	5	3	if (dir:b)=1 then PC<--PC+rel	-	-	-	-	+	-	-	B8 to BF
11	JMP @A	2	1	(PC)<--(A)	-	-	-	-	-	-	-	E0
12	JMP ext	3	3	(PC)<--ext	-	-	-	-	-	-	-	21
13	CALLV #vct	6	1	vector call	-	-	-	-	-	-	-	E8 to EF
14	CALL ext	6	3	subroutine call	-	-	-	-	-	-	-	31
15	XCHW A, PC	3	1	(PC)<--(A), (A)<--(PC)+1	-	-	dH	-	-	-	-	F4
16	RET	4	1	return from subroutine	-	-	-	-	-	-	-	20
17	RETI	6	1	return from interrupt	-	-	-	restore				30

■ Other instructions

Table B.5-4 Other instructions

No.	MNEMONIC	~	#	Operation	TL	TH	AH	N	Z	V	C	OP CODE
1	PUSHW A	4	1		-	-	-	-	-	-	-	40
2	POPW A	4	1		-	-	dH	-	-	-	-	50
3	PUSHW IX	4	1		-	-	-	-	-	-	-	41
4	POPW IX	4	1		-	-	-	-	-	-	-	51
5	NOP	1	1		-	-	-	-	-	-	-	00
6	CLRC	1	1		-	-	-	-	-	-	R	81
7	SETC	1	1		-	-	-	-	-	-	S	91
8	CLRI	1	1		-	-	-	-	-	-	-	80
9	SETI	1	1		-	-	-	-	-	-	-	90

B.6 Instruction map

Table B.6-1 "F²MC-8L instruction map" shows the F²MC-8L instruction map.

■ Instruction map

Table B.6-1 F²MC-8L instruction map

H / L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SWAP	RET	RETI	PUSHW A	POPW A	MOV A, ext	MOVW A, PS	CLRI	SETI	CLRB dir : 0	BBC dir : 0, rel	INCW A	DECW A	JMP @A	MOVW A, PC
1	MULU A	DIVU A	JMP addr16	CALL addr16	PUSHW IX	POPW IX	MOV ext, A	MOVW PS, A	CLRC	SETC	CLRB dir : 1	BBC dir : 1, rel	INCW SP	DECW SP	MOVW SP, A	MOVW A, SP
2	ROLC A	CMP A	ADDC A	SUBC A	XCH A, T	XOR A	AND A	OR A	MOV @A, T	MOV A, @A	CLRB dir : 2	BBC dir : 2, rel	INCW IX	DECW IX	MOVW IX, A	MOVW A, IX
3	RORC A	CMPW A	ADDCW A	SUBCW A	XCHW A, T	XORW A	ANDW A	ORW A	MOVW @A, T	MOVW A, @A	CLRB dir : 3	BBC dir : 3, rel	INCW EP	DECW EP	MOVW EP, A	MOVW A, EP
4	MOV A, #d8	CMP A, #d8	ADDC A, #d8	SUBC A, #d8		XOR A, #d8	AND A, #d8	OR A, #d8	DAA	DAS	CLRB dir : 4	BBC dir : 4, rel	MOVW A, ext	MOVW ext, A	MOVW A, #d16	XCHW A, PC
5	MOV A, dir	CMP A, dir	ADDC A, dir	SUBC A, dir	MOV dir, A	XOR A, dir	AND A, dir	OR A, dir	MOV dir, #d8	CMP dir, #d8	CLRB dir : 5	BBC dir : 5, rel	MOVW A, dir	MOVW dir, A	MOVW SP, #d16	XCHW A, SP
6	MOV A, @IX+d	CMP A, @IX+d	ADDC A, @IX+d	SUBC A, @IX+d	MOV @IX+d, A	XOR A, @IX+d	AND A, @IX+d	OR A, @IX+d	MOV @IX+d,#d8	CMP @IX+d,#d8	CLRB dir : 6	BBC dir : 6, rel	MOVW A, @IX+d	MOVW @IX+d, A	MOVW IX, #d16	XCHW A, IX
7	MOV A, @EP	CMP A, @EP	ADDC A, @EP	SUBC A, @EP	MOV @EP, A	XOR A, @EP	AND A, @EP	OR A, @EP	MOV @EP#, d8	CMP @EP#, d8	CLRB dir : 7	BBC dir : 7, rel	MOVW A, @EP	MOVW @EP, A	MOVW EP, #d16	XCHW A, EP
8	MOV A, R0	CMP A, R0	ADDC A, R0	SUBC A, R0	MOV R0, A	XOR A, R0	AND A, R0	OR A, R0	MOV R0, #d8	CMP R0, #d8	SETB dir : 0	BBS dir : 0, rel	INC R0	DEC R0	CALLV #0	BNC rel
9	MOV A, R1	CMP A, R1	ADDC A, R1	SUBC A, R1	MOV R1, A	XOR A, R1	AND A, R1	OR A, R1	MOV R1, #d8	CMP R1, #d8	SETB dir : 1	BBS dir : 1, rel	INC R1	DEC R1	CALLV #1	BC rel
A	MOV A, R2	CMP A, R2	ADDC A, R2	SUBC A, R2	MOV R2, A	XOR A, R2	AND A, R2	OR A, R2	MOV R2, #d8	CMP R2, #d8	SETB dir : 2	BBS dir : 2, rel	INC R2	DEC R2	CALLV #2	BP rel
B	MOV A, R3	CMP A, R3	ADDC A, R3	SUBC A, R3	MOV R3, A	XOR A, R3	AND A, R3	OR A, R3	MOV R3, #d8	CMP R3, #d8	SETB dir : 3	BBS dir : 3, rel	INC R3	DEC R3	CALLV #3	BN rel
C	MOV A, R4	CMP A, R4	ADDC A, R4	SUBC A, R4	MOV R4, A	XOR A, R4	AND A, R4	OR A, R4	MOV R4, #d8	CMP R4, #d8	SETB dir : 4	BBS dir : 4, rel	INC R4	DEC R4	CALLV #4	BNZ rel
D	MOV A, R5	CMP A, R5	ADDC A, R5	SUBC A, R5	MOV R5, A	XOR A, R5	AND A, R5	OR A, R5	MOV R5, #d8	CMP R5, #d8	SETB dir : 5	BBS dir : 5, rel	INC R5	DEC R5	CALLV #5	BZ rel
E	MOV A, R6	CMP A, R6	ADDC A, R6	SUBC A, R6	MOV R6, A	XOR A, R6	AND A, R6	OR A, R6	MOV R6, #d8	CMP R6, #d8	SETB dir : 6	BBS dir : 6, rel	INC R6	DEC R6	CALLV #6	BGE rel
F	MOV A, R7	CMP A, R7	ADDC A, R7	SUBC A, R7	MOV R7, A	XOR A, R7	AND A, R7	OR A, R7	MOV R7, #d8	CMP R7, #d8	SETB dir : 7	BBS dir : 7, rel	INC R7	DEC R7	CALLV #7	BLT rel

APPENDIX C Mask Options

This appendix lists the mask options for the MB89950/950A series.

■ Mask options

Table C-1 Mask options

No.	Part number	MB89951A MB89953A	MB89P955	MB89PV950
	Specifying procedure	Specify when ordering mask	Set with EPROM programmer	Setting not possible
1	Port pull-up resistor P40 to P46	Can be selected for each pin	Can be selected for each pin	No pull-up resistor
2	Port/Segment output P00 to P07, P10 to P17, P20 to P25	Can be selected for every 8 to 1 pins (*2)	Port/segment output (*3)	Port/segment output (*3)
3	Power-on reset Power-on reset available Power-on reset unavailable	Can be selected	Can be selected	Power-on reset available
4	Selection of main clock oscillation stabilization time (at 5 MHz) (*1) About $2^{18}/F_{CH}$ (about 52.4 ms) About $2^{14}/F_{CH}$ (about 3.28 ms)	Can be selected	Can be selected	$2^{18}/F_{CH}$
5	Reset pin output Reset output available Reset output unavailable	Can be selected	Can be selected	Reset output available

F_{CH}: main clock oscillation frequency

*1: The main clock oscillation stabilization time is generated by dividing the main clock oscillation. Since the oscillation cycle is unstable immediately after oscillation starts, the time in this table is only a guide.

*2: Port/segment output switching should be specified in the same manner as the port allocation set by the segment output select register in the LCD controller described on Chapter 12 "LCD CONTROLLER/DRIVER".

*3: When these pins are used as ports, applied voltage should never be higher than V3.

Table C-2 Recommended port/segment mask option combinations

Number of segments	Number of I/O ports (*1)	Mask Options						
		P00/SEG20 to P07/SEG27	P10/SEG28 to P13/SEG31	P14/SEG32 to P15/SEG33	P16/SEG34	P17/SEG35	P20/SEG36 to P23/SEG39	P24/SEG40 to P25/SEG41
42	11	X	X	X	X	X	X	X
41	12	X	X	X	X	O	X	X
40	13	X	X	X	O	O	X	X
38	15	X	X	O	O	O	X	X
34	19	X	O	O	O	O	X	X
26	27	O	O	O	O	O	X	X
40	13	X	X	X	X	X	X	O
22	31	O	O	O	O	O	O	X
20	33	O	O	O	O	O	O	O

X: Mask option is selected for LCD segment outputs
O: Mask option is selected for port outputs
*1: This column of numbers assume that all the multiplexed peripherals are disabled.

If any customer wants to choose the mask option combination which is not shown in Table C-2 "Recommended port/segment mask option combinations", please inform Fujitsu for special testing arrangement.

APPENDIX D Programming Specifications for One-Time PROM And EPROM Microcontroller

This appendix describes the programming specifications for one-time PROM and EPROM microcontroller.

D.1 "Programming Specifications for One-time PROM and EPROM Microcontrollers"

D.2 "Programming Yield and Erasure"

D.3 "Programming to the EPROM with Piggyback/Evaluation Device"

D.1 Programming Specifications for One-time PROM and EPROM Microcontrollers

In EPROM mode, the MB89P955 function is equivalent to the MBM27C256A. This allows the PROM to be programmed with a general-purpose EPROM programmer by using the dedicated adaptor. Note that the electronic signature mode cannot be used.

■ EPROM programmer socket adaptor

Depending on the EPROM programmer, inserting a capacitor of about 0.1 µF between V_{PP} and V_{SS} or V_{CC} and V_{SS} can stabilize programming operations.

Table D.1-1 "EPROM programmer socket adaptor " lists the EPROM programmer socket adaptors.

Table D.1-1 EPROM programmer socket adaptor

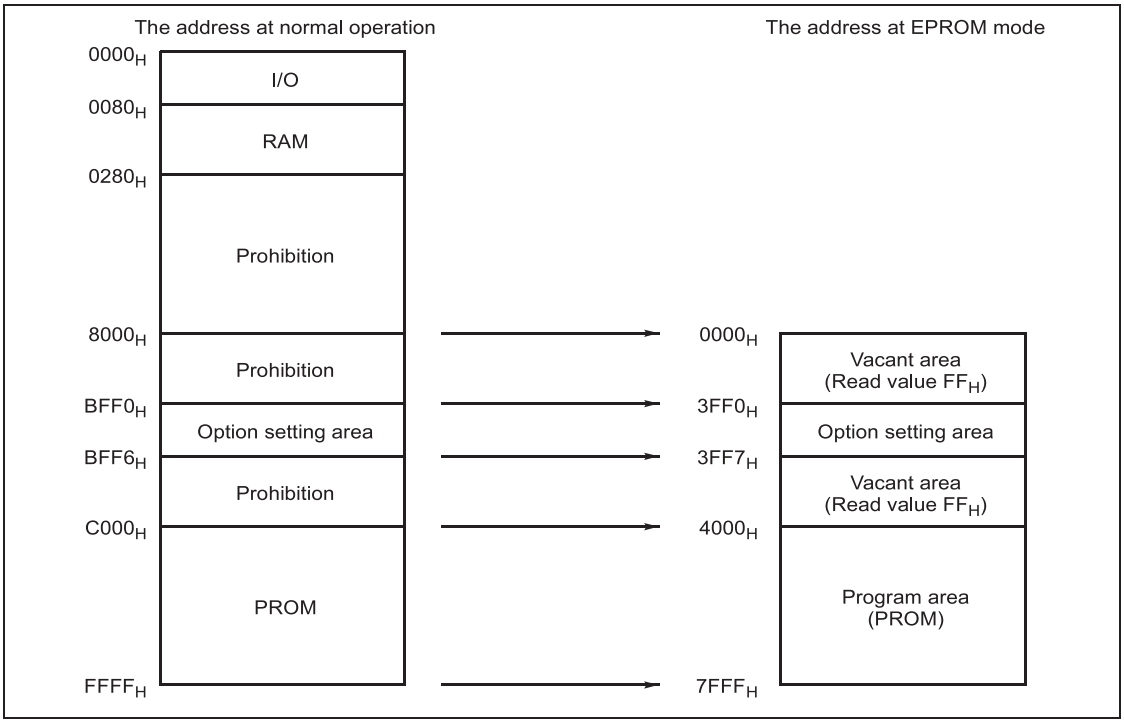
Package	Compatible socket adaptor
FPT-64P-M09	ROM-64QF2-28DP-8L3

Inquiries:
Sun Hayato Co., Ltd. : Phone (81)-3-3986-0403

■ Memory map in EPROM mode

Table D.1-1 "Memory map in EPROM mode" shows the memory map in EPROM mode. Write the option data in the option setting area after consulting the "OTPROM option bit map".

Figure D.1-1 Memory map in EPROM mode

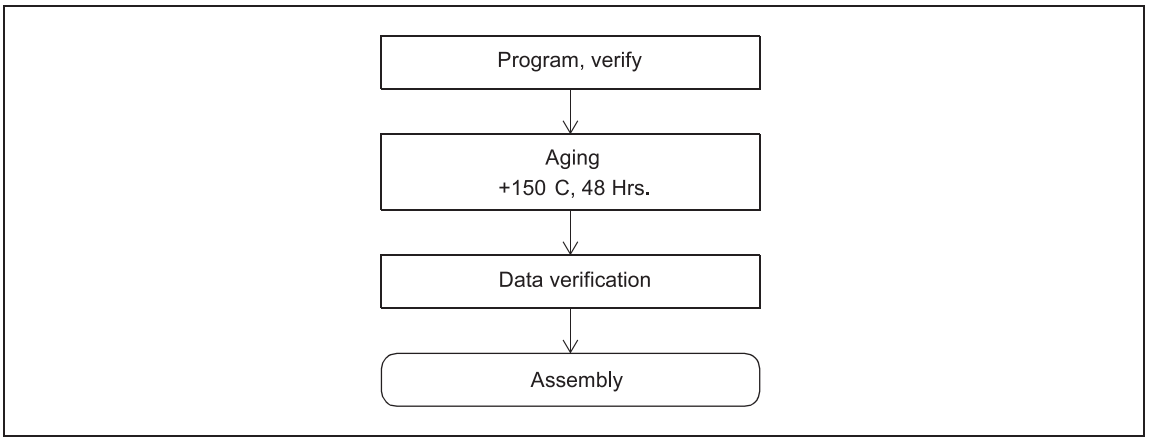


■ Recommended screening conditions

High-temperature aging is recommended as the pre-assembly screening procedure for a product with a blanked OTPROM microcomputer program.

Table D.1-2 "Screening procedure" shows the screening procedure.

Figure D.1-2 Screening procedure



■ Programming to the EPROM

In EPROM mode, the MB89P955 function is equivalent to the MBM27C256A

● Programming procedure:

1. Set the EPROM programmer to the MBM27C256A
2. Load program data from 4000_H to $7FFF_H$ of the EPROM writer (Note that $0C000_H$ to $0FFFF_H$ in the operation mode are equivalent to 4000_H to $7FFF_H$ in the EPROM mode). Load option data from $3FF0_H$ to $3FF6_H$ of the EPROM programmer (See Bit map on the next page for the correspondence to each option).
3. Program with the EPROM programmer.

■ Bit map for PROM option

Table D.1-2 "Bit map for PROM option" shows the bit map for PROM option.

Table D.1-2 Bit map for PROM option

	7	6	5	4	3	2	1	0
$3FF0_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Oscillation stabilization time 1: $2^{18}/F_{CH}$ 0: $2^{14}/F_{CH}$	Reset pin Output 1: Available 0: Unavailable	Power-on Reset 1: Available 0: Unavailable	Vacant Readable/ Writable	Vacant Readable/ Writable
$3FF1_H$	Vacant Readable/ Writable	P46 Pull-up 1: Unavailable 0: Available	P45 Pull-up 1: Unavailable 0: Available	P44 Pull-up 1: Unavailable 0: Available	P43 Pull-up 1: Unavailable 0: Available	P42 Pull-up 1: Unavailable 0: Available	P41 Pull-up 1: Unavailable 0: Available	P40 Pull-up 1: Unavailable 0: Available
$3FF2_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable
$3FF3_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable
$3FF4_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable
$3FF5_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable
$3FF6_H$	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable	Vacant Readable/ Writable

F_{CH}: main clock oscillation frequency

Note:

Initial value is 1 at each bit

D.2 Programming Yield and Erasure

This section describes the programming yield and the data erasure on EPROM microcomputer.

■ Programming yield

All bits cannot be programmed at Fujitsu shipping test to a blanked OTPROM microcomputer, due to its nature. For this reason, a programming yield of 100% cannot be assured at all times.

■ Notes on using and data erasure on EPROM microcomputer

● Erasure

In order to clear all locations of their programmed contents, it is necessary to expose the internal EPROM to an ultraviolet light source. A dosage of 10 W-seconds/cm^2 is required to completely erase an internal EPROM. This dosage can be obtained by exposure to an ultraviolet lamp (wavelength of 2537 Angstroms (Å)) with intensity of 12000 µW/cm^2 for 15 to 21 minutes. The internal EPROM should be about one inch from the source and all filters should be removed from the UV light source prior to erasure.

It is important to note that the internal EPROM and similar devices, will erase with light sources having wavelengths shorter than 4000 Å. Although erasure time will be much longer than with UV source at 2537 Å, nevertheless the exposure to fluorescent light and sunlight will eventually erase the internal EPROM, and exposure to them should be prevented to realize maximum system reliability. If used in such an environment, the package windows should be covered by an opaque label or substance.

D.3 Programming to the EPROM with Piggyback/Evaluation Device

This section describes the programming to the EPROM with piggyback/evaluation device.

■ EPROM for use

MBM27C256A-20TV

■ Programming socket adaptor

To program to the PROM using an EPROM programmer, use the socket adaptor (manufacturer: Sun Hayato Co., Ltd.) listed below.

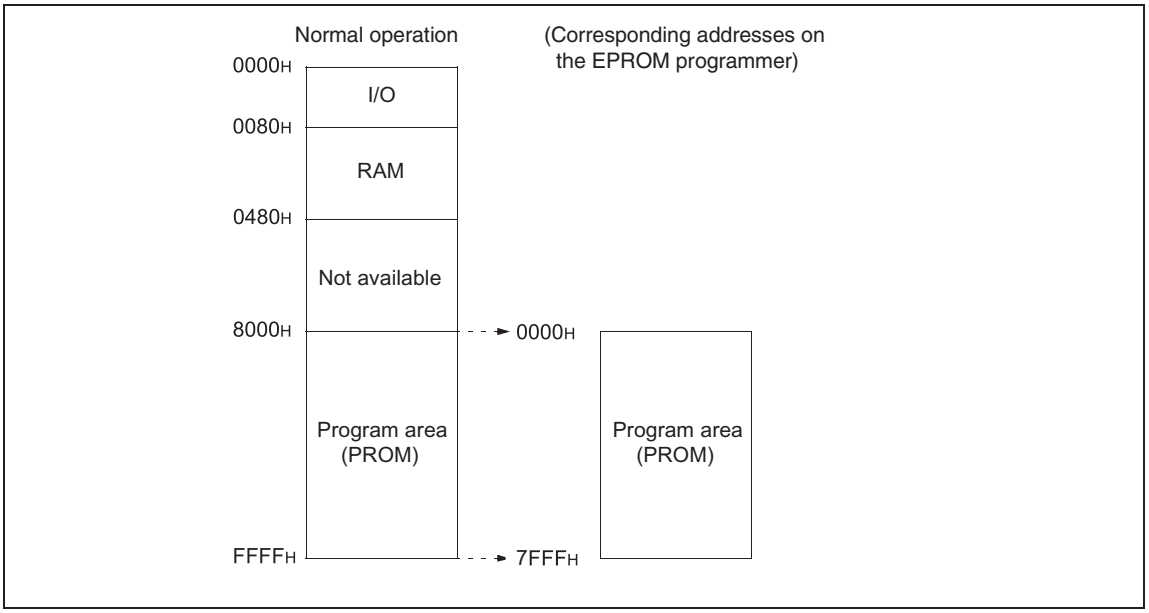
Table D.3-1 Programming socket adaptor

Package	Adaptor socket part number
LCC-32 (Rectangle)	ROM-32LC-28DP-YG

Inquiries:
Sun Hayato Co., Ltd. : Phone (81)-3-3986-0403

■ Memory space

Figure D.3-1 Memory map of piggyback/evaluation device



■ Programming to EPROM

1. Set the EPROM programmer to the MBM27C256A.
2. Load program data into the EPROM programmer at 0000_H to $7FFF_H$.
3. Program to 0000_H to $7FFF_H$ with the EPROM programmer.

APPENDIX E MB89950/950A Series Pin States

This section describes the pin states of the MB89950/950A series in various modes.

■ MB89950/950A series pin states

The state of each pin of the MB89950/950A series of microcontrollers at sleep, stop and reset is as follows:

1. Sleep: The pin state immediately before entering sleep mode is held.
2. Stop: The pin state immediately before entering stop mode is held when the stop mode is started and bit 5 of the standby control register (STBC) is set to "0". When this bit is "1", outputs go to Hi-Z (High impedance) and input/output pins go High.
3. Reset: All I/O go to Hi-Z and peripheral pins (excluding pins for pull-up option) go "H" level.

■ Pin states in various modes

Table E-1 Pin states in various modes

Pin name	Normal operation	Sleep mode	Stop mode (SPL = "0")	Stop mode (SPL = "1")	During reset
COM0 to COM3	COM outputs	COM outputs	Low level outputs	Low level outputs	Low level outputs
SEG0 to SEG19	Segment outputs	Segment outputs	Low level outputs	Low level outputs	Low level outputs
P00/SEG20 to P07/SEG27 P10/SEG28 to P17/SEG35 P20/SEG36 to P25/SEG41	Port I/O / Peripheral output	Port I/O / Peripheral output	Port I/O / Peripheral output = Low	High impedance (*1) / Peripheral output = Low	High impedance (*1)
X0	Input for oscillation	Input for oscillation	High impedance (*1)	High impedance (*1)	Input for oscillation
X1	Output for oscillation	Output for oscillation	High output	High output	Output for oscillation
MODA	Mode input	Mode input	Mode input	Mode input	Mode input
$\overline{RST}$	Reset input	Reset input	Reset input	Reset input	Reset input (*2)
P30, P31	Port I/O	Port I/O	Port I/O	High impedance (*1)	High impedance
P32/V1, P33/V2	Port/LCD bias	Port/LCD bias	Port/LCD bias	High impedance (*4) / LCD bias	High impedance (*5)
V3	Input	Input	Input	Input	Input
P40 to P46/INT0	Port I/O /Peripheral I/O	Port I/O /Peripheral I/O	Port I/O /Peripheral I/O	High impedance (*1, *3)	High impedance (*1)

*1: The internal input level is fixed to prevent leakage due to open input. Pins for which the pull-up option is selected, enter the pull-up state.

*2: The reset pin may serve as the output depending on the option setting.

*3: For P42 and P46, when edge detection for the external interrupt is selected, only the external interrupt can be input even in the stop mode (SPL = "1").

*4: Whether the pins behave as I/O port or LCD bias depends on the PSEL bit of LCDR (see Chapter 12 "LCD CONTROLLER/ DRIVER").

*5: These pins are selected as LCD bias after reset. To turn P32 and P33 to ports after reset, set PSEL bit of LCDR to "1" afterwards.

INDEX

The index follows on the next page.
This is listed in alphabetic order.

Index

Numerics

1/2 bias, 1/2 duty output waveform 251
1/3 bias, 1/3 duty output waveform 254
1/3 bias, 1/4 duty output waveform 257
8-bit PWM timer interrupt source..... 127
8-bit PWM timer interrupt, register and vector table for 131
8-bit PWM timer pin..... 126
8-bit PWM timer pin, block diagram of 126
8-bit PWM timer registers..... 127
8-bit PWM timer, block diagram of 124
8-bit PWM timer, note on using..... 137
8-bit serial I/O interrupt source 175
8-bit serial I/O Pins 173
8-bit serial I/O pins, block diagram of 174
8-bit serial I/O registers 174
8-bit Serial I/O, block diagram of..... 171
8-bit serial I/O, note on using 188

A

addressing, explanation of 269
arithmetic instruction 281
arithmetic operation result bits 29

B

bi-directional serial I/O performing 189
bit manipulation instruction, read operation upon execution of..... 278
branch instruction..... 284

C

clock controller, block diagram of..... 53
clock generator..... 51
clock supply function 101
clock supply function, operation of 107
clock supply map..... 49
condition code register (CCR), structure of..... 29

D

dedicated register configuration 27
dedicated register functions 27
differences among products..... 6
display brightness adjustment when internal voltage divider is used 238
display RAM and output pin 248

E

effect of reset on RAM content 47
EPROM for use..... 294
EPROM microcomputer, note on using and data erasure on..... 293
EPROM mode, memory map in 290
EPROM programmer socket adaptor..... 290
EPROM, programming to..... 291, 294
external interrupt circuit interrupt source..... 227
external interrupt circuit interrupt, register and vector table for..... 230
external interrupt circuit pin..... 226
external interrupt circuit pin, block diagram of 226
external interrupt circuit register..... 227
external interrupt circuit, block diagram of 225
external interrupt circuit, functions of 224
external interrupt circuit, operation of..... 231
external interrupt circuit, program example for 232
external interrupt control register (EIC)..... 228
external reset pin function..... 45
external reset pin, block diagram of..... 45
external shift clock, using..... 186
external voltage divider..... 239
external voltage divider, use of 240

F

F^2MC-8L instruction, overview of..... 267
FPT-64P-M09 package dimension 10
FPT-64P-M09 pin assignment 8

G

general-purpose register areas (addresses: 0100H to 01FFH)..... 24
general-purpose registers, features of..... 34
general-purpose registers, structure of..... 33

H

handling device, note on..... 18

I

I/O map 264
I/O pins and pin functions 12

I/O port function 70
I/O port, program example for 98
instruction cycle 54
instruction map 286
instruction, symbol used with 268
internal shift clock, using 185
internal voltage divider 237
internal voltage divider, use of 238
interrupt acceptance control bit 30
interrupt level setting registers (ILR1, ILR2, ILR3), structure of 36
interrupt processing 37
interrupt processing time 40
interrupt processing, stack area for 42
interrupts when external interrupt circuit is operating 230
interval timer function 100, 142
interval timer function (one-shot timer mode), program example 2 for 167
interval timer function (reload timer mode), program example 1 for 165
interval timer function (square wave output function) 122
interval timer function (timebase timer), operation of 107
interval timer function, interrupt for 106, 131, 155
interval timer function, operation of 132, 156
interval timer function, program example for 138

L

LCD control register (LCDR) 244
LCD controller/driver function 234
LCD controller/driver pin 241
LCD controller/driver pin, block diagram of 242
LCD controller/driver RAM 243
LCD controller/driver register 243
LCD controller/driver, block diagram of 235
LCD controller/driver, operation of 250
LCD controller/driver, program example for 260
LCD driving waveform 250

M

main clock oscillation stabilization delay time 55
main clock oscillation stabilization delay time and reset source 44
mask option 287
MB89950/950A series block diagram 7
MB89950/950A series pin state 295
measuring long pulse widths 160
memory access mode selection operation 68
memory map 23
memory space 294
memory space structure 22
mode data 67
mode fetch 47
mode pin 47
mode pin (MODA) 67
MQP-64C-P01 package dimension 11
MQP-64C-P01 pin assignment 9
multiple interrupts 39

N

note when changing edge polarity selection 230

O

operation during standby mode or operation halt 135, 163
operation mode 0, 1, 3, operation of 217
oscillation stabilization delay reset state 47
oscillation stabilization delay time 55, 66, 106
other instruction 285

P

peripheral function, interrupt request from 35
pin state in various mode 295
pin states after reading mode data 48
pin states during reset 48
port 0 data register function 74
port 0 pin, block diagram of 73
port 0 pins 72
port 0 register 73
port 0, operation of 75
port 0, structure of 72
port 1 data register function 79
port 1 pin, block diagram of 78
port 1 pins 77
port 1 register 78
port 1, operation of 80
port 1, structure of 77
port 2 data register function 84
port 2 pin, block diagram of 83
port 2 pins 82
port 2 register 83
port 2, operation of 85
port 2, structure of 82
port 3 data register function 89
port 3 pin, block diagram of 87

port 3 pins 86
port 3 register 88
port 3, operation of 90
port 3, structure of 86
port 4 pin, block diagram of 93
port 4 pins 92
port 4 register function 94
port 4 registers 93
port 4, operation of 96
port 4, structure of 92
product range, MB89950/950A series 4
programming socket adaptor 294
programming yield 293
PROM option, bit map for 292
pulse width count timer interrupt source 147
pulse width count timer interrupt, register and vector table for 155
pulse width count timer pin 146
pulse width count timer pin, block diagram of 146
pulse width count timer registers 147
pulse width count timer, block diagram of 144
pulse width count timer, note on using 164
pulse width measurement function 143
pulse width measurement function, interrupt for 155
pulse width measurement function, operation of 159, 162
PWC noise filter control register (NCCR) 154
PWC pulse width control register 1 (PCR1) 148
PWC pulse width control register 2 (PCR2) 150
PWC reload buffer register (RLBR) 152
PWM compare register (COMR) 130
PWM control register (CNTR) 128
PWM timer function 123
PWM timer function, operation of 134
PWM timer function, program example for 140

R

read-modify-write operation 278
receive interrupt 215
receive operation 218
recommended screening condition 291
register bank pointer (RP), structure of 32
reset operation, overview of 46
reset source 43, 44

S

segment output select register (SEGR) 246
serial data register (SDR) 179
serial I/O function 170
serial input data register (SIDR) 211
serial input operation 183
serial input, operation at completion of 184
serial input, program example for 192
serial mode control register 1 (SMC1) 205
serial mode control register 2 (SMC2) 213
serial mode register (SMR) 176
serial output data register (SODR) 212
serial output operation 181
serial output operation, interrupt for 180
serial output, operation at completion of 182
serial output, program example for 190
serial rate control register (SRC) 207
serial status and rate register (SSD) 209
single-chip mode 67
sleep mode, operation of 59
special instruction 274
stack operation at interrupt return 41
stack operation at start of interrupt processing 41
standby control register (STBC) 61
standby mode 57
standby mode and interrupts, go to 65
standby mode by interrupt, wake-up from 65
standby mode, note on setting 65
standby mode, operating states during 58
state transition diagram 63
stop mode, operation of 60
storing 16-bit data in RAM 26
storing 16-bit data on stack 26
storing 16-bit operands 26

T

timebase timer control register (TBTC) 104
timebase timer interrupt 106
timebase timer interrupt, register and vector table for 106
timebase timer, block diagram of 102
timebase timer, note on using 109
timebase timer, operation of 108
timebase timer, program example for 110
transfer clock, slection of 195
transfer instruction 279
transmit interrupt 215
transmit operation 217

U

UART function 194
UART interrupt, register and vector table for 215
UART pin 202

UART pin, block diagram of 203
UART registers 204
UART, block diagram of 199
UART, operation of 216
UART, program example for 220

V

vector table area (addresses: FFC0H to FFFFH) .. 25

W

watchdog timer control register (WDTC) 115
watchdog timer function.....112
watchdog timer, block diagram of.....113
watchdog timer, note on using.....118
watchdog timer, operation of 116
watchdog timer, program example for.....119

CM25-10146-1E

FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL

F^2MC-8L
8-BIT MICROCONTROLLER
MB89950/950A Series
HARDWARE MANUAL

July 2002 the first edition

Published **FUJITSU LIMITED** Electronic Devices

Edited Technical Information Dept.